A High Order Collocation Based Nystrom Scheme for Second Order Initial Value problems

Dissertation submitted to

Visvesvaraya National Institute of Technology, Nagpur

in partial fulfillment of the requirement for the award of degree

Master of Science

in

Mathematics

by

Nayan Bhagchand Adwani

(MS23MTH003)

under the guidance of

Dr. P. Pramod Chakravarthy

Department of Mathematics

Visvesvaraya National Institute of Technology

Nagpur 440010 (India)

2025

A High Order Collocation Based Nystrom Scheme for Second Order Initial Value problems

Dissertation submitted to

Visvesvaraya National Institute of Technology, Nagpur

in partial fulfillment of the requirement for the award of degree

Master of Science

in

Mathematics

by

Nayan Bhagchand Adwani

(MS23MTH003)

under the guidance of

Dr. P. Pramod Chakravarthy

Department of Mathematics

Visvesvaraya National Institute of Technology

Nagpur 440010 (India)

2025

©Visvesvaraya National Institute of Technology (VNIT) 2009

Department of Mathematics

Visvesvaraya National Institute of Technology, Nagpur

Declaration

I, Nayan Bhagchand Adwani, hereby declare that the dissertation titled "**A High Order Collocation Based Nystrom Scheme for Second Order Initial Value problems**" has been conducted by me in the Department of Mathematics at Visvesvaraya National Institute of Technology, Nagpur. This work is original and has not been submitted, either in whole or in part, for any degree or diploma at this or any other institution.

Nayan Bhagchand Adwani
Date:

Certificate

This is to certify that the dissertation titled "**A High Order Collocation Based Nystrom Scheme for Second Order Initial Value problems**" is submitted by **Nayan Bhagchand Adwani** in partial fulfillment of the requirements for the award of the degree of **Master of Science in Mathematics**, VNIT Nagpur. The work is thorough, complete, and ready for final assessment.

Dr. P. Pramod Chakravarthy
Professor, Department of Mathematics
VNIT, Nagpur

Head, Department of Mathematics
VNIT, Nagpur
Date:

Non - Plagiarism Certificate

Certified that the M.Sc. dissertation titled **"A High Order Collocation Based Nystrom Scheme for Second Order Initial Value problems"** submitted by **Nayan Bhagchand Adwani**, Enrollment No. **MS23MTH003**, ID No. **30094** has been checked for plagiarism using TURNITIN software, and the overlap is found within prescribed limits.

A summary of the plagiarism report is given below:

ORIGINALITY REPORT

18%	15%	14%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

Signature:

Student Name: Nayan Bhagchand Adwani

Enrollment No: MS23MTH003

ID No: 30094

Dr. P. Pramod Chakravarthy

Professor, Department of Mathematics

VNIT, Nagpur

Acknowledgment

It is with great pleasure and pride that I express my deep sense of gratitude to my supervisor, Dr. P. Pramod Chakravarthy, Professor, Department of Mathematics, Visvesvaraya National Institute of Technology, Nagpur. I am immensely grateful for his vital encouragement and guidance throughout my project work. Without his active support, completing this work would not have been possible.

I convey my deep sense of gratitude to Dr. Pallavi Mahale Professor and Head of the Department of Mathematics at VNIT Nagpur, for her valuable cooperation and the excellent facilities provided throughout the session. I also extend my thanks to the project progress and evaluation committee members, Dr. Pradip Roul and Dr. Vijender Nallapu, for their encouragement during the progress of my work. I am sincerely thankful to all the faculty members of the Department of Mathematics at VNIT Nagpur for their continuous support. My sincere appreciation goes to Dr. Vishwas Sundrani and the Ph.D. scholars Mr. Sachin Kumar Sahu and Mr. Rahul Mishra in the Department of Mathematics for their assistance and complete support.

I am thankful to all my friends for their kind cooperation and to my family for their inspiration and support. Lastly, I express my gratitude to everyone who helped me directly or indirectly during the completion of this project.

Nayan Bhagchand Adwani

List of Notations

We are using the subsequent standard notations and symbols without defining them explicitly:

Symbol	**Meaning**
$\in$	belongs to
$\forall$	for all
$\Rightarrow$	implies
$\rightarrow$	maps to
$\subset$	subset of
$\mathbb{N}$	set of natural numbers
$\mathbb{R}$	set of real numbers
$\leq$	less than or equal to
$y(x)$	exact solution of the initial value problem (IVP)
$y'(x), y''(x), \ldots, y^{(n)}(x)$	n^{th} derivative of $y(x)$
h	step size
x_0	initial point
$x_1 = x_0 + h$	next grid point
c_j	collocation node, $j = 1, 2, 3, 4$
$f(x, y, y')$	function defining the second-order ODE
f_{c_j}	second derivative evaluated at $x_0 + c_j h$
$\alpha_j(s)$	weight function in position approximation
$\beta_j(s)$	weight function in derivative approximation
y_{c_j}	numerical solution at collocation point $x_0 + c_j h$
y'_{c_j}	numerical derivative at collocation point $x_0 + c_j h$
y_1	numerical solution at $x_1 = x_0 + h$
y'_1	numerical derivative at $x_1 = x_0 + h$
LTE	local truncation error

MAE	maximum absolute error
$\Gamma[y(x);h]$	operator representing the algebraic error in the method
$O(h^n)$	order of the error term
p	order of accuracy
V_o, V_o', F_o	vector matrices of function values, derivatives, and second derivatives
$\bar{R}, \bar{S}, \bar{T}$	matrices containing coefficients in matrix form of the Nystrom method

Contents

1 Introduction to Multistep Methods **1**

1.1 Multistep Methods 1

1.2 Nystrom Methods 4

1.3 Numerical Results 6

2 Collocation-Based Nystrom Scheme for Second-Order Initial Value Problems **8**

2.1 Derivation of the Method 9

2.2 Local Trunction Error 14

2.3 Overview of the Developed Scheme 22

3 Numerical Experiments **23**

3.1 Numerical Results (2nd-order linear IVPs) 25

3.2 Numerical results (2nd-order non-linear IVPs) 30

4 Conclusion **36**

References **37**

Abstract

This thesis presents the development and analysis of A High Order Collocation Based Nystrom Scheme for Second Order Initial Value Problems, applicable to both linear and nonlinear differential equations. Motivated by the work done in [2], we studied the collocation Nystrom method for solving the initial value problems. The work begins with an overview of classical multistep methods, including the Nyström family, to establish a theoretical foundation for the proposed approach. The method is based on a high-order Taylor series expansion and strategically chosen collocation points, which enable the minimization of local truncation errors and enhancement of numerical accuracy.

By determining optimal coefficients, the proposed scheme achieves a global step point order of 8, ensuring high precision even for small step sizes. The formulation leverages techniques to balance computational efficiency with accuracy, making it suitable for a wide range of scientific and engineering applications.

Extensive numerical experiments were conducted on various test problems to evaluate the performance of the method. The results confirm the robustness, reliability, and effectiveness of the proposed approach in producing accurate solutions with minimal errors. This work also lays a foundation for future studies involving further theoretical advancements and adaptive strategies in the numerical treatment of second-order IVPs.

Chapter 1

Introduction to Multistep Methods

Numerical methods for solving differential equations have long played a central role in the mathematical modeling of real-world systems across disciplines such as physics, engineering, biology, and economics. Among the various types of differential equations, second-order initial value problems (IVPs) frequently arise in mechanical vibrations, electrical circuits, and wave phenomena.

Analytical solutions to such equations are often unavailable, especially when the underlying models are nonlinear or involve complex boundary or initial conditions. As a result, it becomes essential to rely on robust and accurate numerical methods to approximate their solutions.

One such family of numerical techniques is the class of multistep methods, which includes the well-known Nystrom methods. These methods utilize multiple past values to achieve higher-order accuracy with reduced computational cost per step compared to single-step methods. When appropriately optimized, they can provide exceptional performance even for challenging problems.

In this chapter, we begin with a review of classical multistep methods and the foundational concepts that support them. This lays the groundwork for the optimized strategy we propose later in this work.

1.1 Multistep Methods

To numerically solve the initial value problem

$$u' = f(t, u), \quad u(t_0) = u_0, \tag{1.1.1}$$

we may use multistep methods, also known as *k-step methods*. These approaches compute the next value $u(t_{j+1})$ using previously evaluated points of $u(t)$ and possibly its derivatives at multiple earlier mesh points $t_j, t_{j-1}, \ldots, t_{j-k+1}$, and potentially at t_{j+1} itself. If the method includes the unknown value at t_{j+1} when computing $u(t_{j+1})$, it is referred to as an *implicit method*. If it only uses known values from earlier steps and excludes t_{j+1}, it is called an *explicit method*. A general integral form of such a method is given by:

$$u(t_{j+1}) = u(t_j) + \int_{t_j}^{t_{j+1}} f(t,u)\,dt. \tag{1.1.2}$$

Since the function $u(t)$ is unknown, the integral above cannot be computed exactly. Therefore, we approximate $f(t,u)$ by a suitable polynomial $P(t)$, typically of degree $k-1$, that interpolates f at known mesh points $t_j, t_{j-1}, \ldots, t_{j-k+1}$. This approximation facilitates the development of both explicit and implicit multistep schemes. If the interpolant excludes t_{j+1}, the method remains explicit. Otherwise, the inclusion of f at t_{j+1} leads to an implicit formulation. The Newton backward difference interpolating polynomial interpolating the data (t_j, f_j), $(t_{j-1}, f_{j-1}), \ldots, (t_{j-k+1}, f_{j-k+1})$ is given by:

$$\begin{aligned} P_{k-1}(t) = f_j + \frac{(t-t_j)}{h}\nabla f_j + \frac{(t-t_j)(t-t_{j-1})}{2!h^2}\nabla^2 f_j + \cdots \\ + \frac{(t-t_j)(t-t_{j-1})\cdots(t-t_{j-k+2})}{(k-1)!h^{k-1}}\nabla^{k-1} f_j \\ + \frac{(t-t_j)(t-t_{j-1})\cdots(t-t_{j-k+1})}{k!} f^{(k)}(\xi) \end{aligned} \tag{1.1.3}$$

where ξ lies in the interval containing the points $t_j, t_{j-1}, \cdots, t_{j-k+1}$ and t.
Substituting $t - t_j = hs$ in (1.1.3), we get

$$\begin{aligned} P_{k-1}(t_j + hs) = f_j + s\nabla f_j + \frac{s(s+1)}{2}\nabla^2 f_j + \cdots + \frac{s(s+1)\ldots(s+k-2)}{(k-1)!}\nabla^{k-1} f_j \\ + \frac{s(s+1)\ldots(s+k-1)}{k!} h^k f^{(k)}(\xi) \end{aligned}$$

where $s = \frac{t-t_j}{h}$ and $h = t_{j+1} - t_j$;

$$P_{k-1}(t_j + hs) = \sum_{m=0}^{k-1} (-1)^m \binom{-s}{m} \nabla^m f_j + (-1)^k \binom{-s}{k} h^k f^{(k)}(\xi) \tag{1.1.4}$$

$$where\binom{-s}{m} = (-1)^m \frac{s(s+1)\cdots(s+m-1)}{m!} \tag{1.1.5}$$

Replacing $f(t,u)$ in (1.1.2) by the interpolating polynomial $P_{k-1}(t_j + hs)$ as given in (1.1.4), we get

$$u(t_{j+1}) = u(t_{j-i}) + h\int_{-i}^{1}\left[\sum_{m=0}^{k-1}(-1)^m\binom{-s}{m}\nabla^m f_j + (-1)^k\binom{-s}{k}h^k f^{(k)}(\xi)\right]ds$$

$$= u(t_{j-i}) + h\sum_{m=0}^{k-1}\gamma_m^{(i)}\nabla^m f_j + T_k^{(i)} \tag{1.1.6}$$

where

$$\gamma_m^{(i)} = \int_{-i}^{1}(-1)^m\binom{-s}{m}ds \tag{1.1.7}$$

and

$$T_k^{(i)} = h^{k+1}\int_{-i}^{1}(-1)^k\binom{-s}{k}f^{(k)}(\xi)ds. \tag{1.1.8}$$

Neglecting the error term $T_k^{(i)}$ in (1.1.6), we obtain the explicit multistep method:

$$u_{j+1} = u_{j-i} + h\sum_{m=0}^{k-1}\gamma_m^{(i)}\nabla^m f_j. \tag{1.1.9}$$

Since by (1.1.8) the local trunction error is of $O(h^{k+1})$, the method (1.1.9) is at least of order k.

From (1.1.5) and (1.1.7), we obtain:

$$\gamma_0^{(i)} = \int_{-i}^{1} ds = 1 + i.$$

$$\gamma_1^{(i)} = \int_{-i}^{1} s\,ds = \frac{1}{2}(1+i)(1-i).$$

$$\gamma_2^{(i)} = \frac{1}{2}\int_{-i}^{1} s(s+1)ds = \frac{1}{12}(5 - 3i^2 + 2i^3).$$

$$\gamma_3^{(i)} = \frac{1}{6}\int_{-i}^{1} s(s+1)(s+2)ds = \frac{1}{24}(3-i)(3+i-i^2+i^3).$$

$$\gamma_4^{(i)} = \frac{1}{24}\int_{-i}^{1} s(s+1)(s+2)(s+3)ds = \frac{1}{720}(251 - 90i^2 + 110i^3 - 45i^4 + 6i^5)$$

and so on. If we express $\nabla^m f_j$ in terms of the function values f_m, we can also write the method (1.1.9) in the form

$$u_{j+1} = u_{j-i} + h\sum_{m=0}^{k-1} \gamma^{*(i)}_m f_{j-m} \tag{1.1.10}$$

For different values of 'i' in (1.1.9) or (1.1.10) we obtain different methods:

For $i = 0$; we have Adam's Bashforth Methods.

For $i = 1$; we have Nystrom Methods.

1.2 Nystrom Methods

In this section, we apply the Nystrom strategy to first-order IVPs to explore its polynomial-based structure and assess its truncation error behavior. This foundational study also provides insights into the polynomial framework that will later be extended and optimized for second-order problems.

On substituting the value of $\gamma_m^{(1)}$ in (1.1.9), we get

$$u_{j+1} = u_{j-1} + h\left[2f_j + 0 \times \nabla f_j + \frac{1}{3}\nabla^2 f_j + \frac{1}{3}\nabla^3 f_j + \frac{29}{90}\nabla^4 f_j + \cdots\right] \tag{1.2.1}$$

Replacing the differences by their function values, we obtain

$$u_{j+1} = u_{j-1} + h\sum_{m=0}^{k-1} \gamma^{*(1)}_m f_{j-m} \tag{1.2.2}$$

The coefficients $\gamma_m^{*(1)}$ are listed in the following table:

k	$\gamma_0^{*(1)}$	$\gamma_1^{*(1)}$	$\gamma_2^{*(1)}$	$\gamma_3^{*(1)}$	$\gamma_4^{*(1)}$
1	2				
2	2	0			
3	$\frac{7}{3}$	$-\frac{2}{3}$	$\frac{1}{3}$		
4	$\frac{8}{3}$	$-\frac{5}{3}$	$\frac{4}{3}$	$-\frac{1}{3}$	
5	$\frac{269}{90}$	$-\frac{266}{90}$	$\frac{294}{90}$	$-\frac{146}{90}$	$\frac{29}{90}$

From $(1.1.8)$, the trunction error is given by:

$$T_k^{(1)} = h^{k+1} \int_{-1}^{1} g(s) f^{(k)}(\xi) ds$$

where

$$g(s) = [s(s+1)\cdots(s+k-1)]/k! \qquad \text{(from 1.1.5)}$$

Since $g(s)$ changes sign in $(-1, 1)$, the mean value theorem cannot be applied. However, a bound for the error can be written as

$$|T_k^{(1)}| \leq h^{k+1} M_k \int_{-1}^{1} |g(s)| ds$$

where

$$M_k = \max_{-1 \leq t \leq 1} |f^{(k)}(t)|.$$

Note that since $\gamma_1^{*(1)} = 0$, the formulas corresponding to $k = 1$ and $k = 2$ are the same. That is, the truncation error of the formula

$$u_{j+1} = u_{j-1} + 2hf_j$$

is of $O(h^3)$ and the method is of second order. This method is the same as the mid-point method.
For $k = 3$, the truncation error of the formula

$$u_{j+1} = u_{j-1} + \frac{h}{3}(7f_j - 2f_{j-1} + f_{j-2})$$

is of $O(h^4)$ and the method is of third order.
For $k = 4$, the truncation error of the formula

$$u_{j+1} = u_{j-1} + \frac{h}{3}(8f_j - 5f_{j-1} + 4f_{j-2} - f_{j-3})$$

is of $O(h^5)$ and the method is of fourth order.
For $k = 5$, the truncation error of the formula

$$u_{j+1} = u_{j-1} + \frac{h}{90}(269f_j - 266f_{j-1} + 294f_{j-2} - 146f_{j-3} + 29f_{j-4})$$

is of $O(h^6)$ and the method is of fifth order.

The application of the Nystrom method to first-order problems helps illustrate the underlying polynomial approximation principles that will later be leveraged for second-order equations. The analysis that follows focuses on the method's local truncation error and sets the stage for further refinement.

1.3 Numerical Results

Example 1.1. Given an initial value problem:

$$y' = -2x - y, \quad y(0) = -1$$

with a step size $h = 0.2$ using the Nystrom third-order method compute y(0.6).
Solution: Here $x_0 = 0, \quad x_1 = 0.2, \quad x_2 = 0.4, \quad x_3 = 0.6, \quad y_0 = -1.$

$$y_3 = y_1 + \frac{h}{3}(7f_2 - 2f_1 + f_0) \tag{1.3.1}$$

where $f_j = -2x_j - y_j; \quad (j = 0, 1, 2)$
for y_3 we need to find f_0, f_1, f_2 also y_1, y_2.
for which we will use Euler method; $y_{j+1} = y_j + hf(x_0, y_0)$
We get $\quad f_0 = 1, \quad y_1 = -0.8, \quad f_1 = 0.4, \quad y_2 = -0.72, \quad f_2 = -0.08$
thus from equation (1.3.1) we get, $y_3 = -0.8240$
The exact solution to the IVP is: $\quad y(x) = -2x - 2 - 3e^{-x}$
Hence, $y_3 = y(0.6) = -0.8464$
Thus we get, $error = |-0.8464 - (-0.8240)| = 0.0224$

Example 1.2. Using the Nystrom fourth order method solve the initial value problem

$$u' = -2tu^2, \quad u(0) = 1 \quad \text{on } [0, 2], \text{ with } h = 0.2.$$

Solution: Here $x_0 = 0,\ x_1 = 0.2,\ x_2 = 0.4,\ \cdots,\ x_9 = 1.8,\ x_{10} = 2.$

$$y_4 = y_2 + (h/3)(8f_3 - 5f_2 + 4f_1 - f_0) \tag{1.3.2}$$

where, $f_j = -2x_j {y_j}^2; \quad (j = 0, 1, 2, 3)$

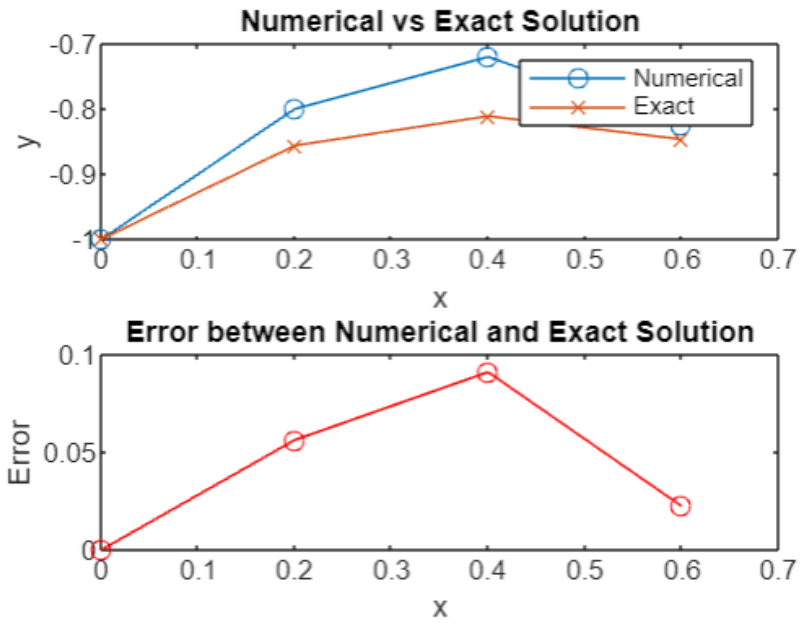


Figure 1.1: Numerical solution and error for problem 1.

$y_4,\ y_5,\ y_6,\ y_7,\ y_8,\ y_9,\ y_{10}$ we need to find $y_1,\ y_2,\ y_3$.

Initially for $y_1,\ y_2,\ y_3$, we will use Runge-Kutta 4th order method;

We get, $y_1 = 0.9615$, $y_2 = 0.8621$, $y_3 = 0.7353$

thus from equation (1.3.2) we get, $y_4 = 0.6156$, $y_5 = 0.4943$,

$y_6 = 0.4237$, $y_7 = 0.3089$, $y_8 = 0.3351$, $y_9 = 0.1239$, $y_{10} = 0.3828$

Thus, $y(2) \approx y_{10} = 0.3828$.

For the detailed discussion on Multistep methods, one can see the references [1].

Remark: The methods discussed so far primarily address first-order initial value problems. However, many physical systems such as mechanical oscillations, electrical circuits, and wave propagation are naturally modeled using second-order differential equations. Although it is possible to reduce such equations to first-order systems, doing so often leads to unnecessary complexity and computational overhead.

To address second-order IVPs more directly and efficiently, we now turn our attention to specialized methods that are inherently designed for such problems. In the next chapter, we present the derivation of an collocation-based Nystrom scheme, specifically constructed to solve second-order IVPs with high accuracy and minimal error propagation.

Chapter 2

Collocation-Based Nystrom Scheme for Second-Order Initial Value Problems

Introduction:

While classical multistep methods and Nystrom-type formulas provide effective numerical solutions to second-order differential equations, their accuracy and efficiency are often limited by fixed step sizes and static error control. In particular, traditional Nystrom methods may suffer from the accumulation of local truncation errors and reduced precision when applied to stiff or non-linear problems.

Moreover, increasing the order of such methods typically requires additional derivative evaluations or a larger number of mesh points, which may not be computationally desirable. To overcome these limitations, it becomes essential to design methods that not only improve the local and global error behavior but also adapt to the structure of the underlying differential system.

This forms the primary motivation behind the construction of an *collocation-based Nystrom scheme* a high-order technique that leverages carefully selected collocation points and Taylor expansions to systematically reduce truncation errors. By integrating it into the method design, we aim to develop a numerical solver that offers both precision and efficiency, particularly for second-order IVPs with smooth solutions.

In this chapter, we derive such a method in detail, starting from polynomial approximations and error analysis, and culminating in a practical algorithm for both linear and non-linear second-order problems.

2.1 Derivation of the Method

We consider second-order initial value problems (IVPs) of the general form

$$y''(x) = f(x, y(x), y'(x)), \quad y(a) = y_0, \quad y'(a) = y'_0, \quad x \in [a, b] \subset \mathbb{R}, \tag{2.1.1}$$

where $a = x_0$ stands for the initial point, the function f is assumed to be sufficiently smooth on the interval of interest. Such equations commonly arise in physical systems governed by Newton's laws, wave propagation, and other dynamic processes.

Traditional approaches often reduce a second-order ODE to a system of first-order equations before applying numerical solvers. However, this reduction can introduce additional computational complexity and error propagation. Therefore, it is desirable to design methods that directly handle second-order equations while retaining high accuracy.

To derive a high-order Nystrom-type method, we assume that the exact solution $y(x)$ can be approximated locally on each subinterval $[x_0, x_1]$, where $x_1 = x_0 + h$, by a polynomial $u(x)$ of degree 7. This polynomial is constructed to satisfy second-derivative conditions at carefully selected collocation points. Specifically, we impose that

$$u''(x_{c_j}) = f_{c_j}; \quad (for\ j = 0, 1, \ldots, 5), \tag{2.1.2}$$

where $0 = c_0 < c_1 < c_2 < c_3 < c_4 < c_5 = 1$ are collocation points chosen to optimize the accuracy of the method, $x_{c_j} = x_0 + c_j h; \quad (for\ j = 0, 1, 2, 3, 4, 5)$.
We now approximate the second derivative $u''(x_0+sh)$ using a Lagrange interpolating polynomial through these collocation points. Let $s \in [0, 1]$ be a normalized local variable. Then,

$$u''(x_0 + sh) \approx \sum_{j=0}^{5} f_{c_j} L_j(s), \tag{2.1.3}$$

where $f_{c_j} = f(x_{c_j}, u(x_{c_j}), u'(x_{c_j}))$ and $L_j(s)$ are the Lagrange basis polynomials defined over the collocation nodes such as:

$$L_j(s) = \prod_{\substack{0 \le m \le 5 \\ m \ne j}} \frac{s - c_m}{c_j - c_m}$$

The Lagrange basis polynomial $L_j(s)$ for $j = 0, 1, 2, 3, 4, 5$ is:

$$L_0(s) = \frac{(s-c_1)(s-c_2)(s-c_3)(s-c_4)(s-c_5)}{(c_0-c_1)(c_0-c_2)(c_0-c_3)(c_0-c_4)(c_0-c_5)},$$

$$L_1(s) = \frac{(s-c_0)(s-c_2)(s-c_3)(s-c_4)(s-c_5)}{(c_1-c_0)(c_1-c_2)(c_1-c_3)(c_1-c_4)(c_1-c_5)},$$

$$L_2(s) = \frac{(s-c_0)(s-c_1)(s-c_3)(s-c_4)(s-c_5)}{(c_2-c_0)(c_2-c_1)(c_2-c_3)(c_2-c_4)(c_2-c_5)},$$

$$L_3(s) = \frac{(s-c_0)(s-c_1)(s-c_2)(s-c_4)(s-c_5)}{(c_3-c_0)(c_3-c_1)(c_3-c_2)(c_3-c_4)(c_3-c_5)},$$

$$L_4(s) = \frac{(s-c_0)(s-c_1)(s-c_2)(s-c_3)(s-c_5)}{(c_4-c_0)(c_4-c_1)(c_4-c_2)(c_4-c_3)(c_4-c_5)},$$

$$L_5(s) = \frac{(s-c_0)(s-c_1)(s-c_2)(s-c_3)(s-c_4)}{(c_5-c_0)(c_5-c_1)(c_5-c_2)(c_5-c_3)(c_5-c_4)},$$

where s is the variable and $c_0,\ c_1,\ c_2,\ c_3,\ c_4,\ c_5$ are the interpolation points.
Integrating (2.1.3) once and twice, respectively, and exploiting the initial conditions for $s = 0$ leads to the following formulas:

$$y'(x_0 + sh) \simeq u'(x_0 + sh) = y'_0 + h\sum_{j=0}^{5} \beta_j(s) f_{c_j}, \tag{2.1.4}$$

and

$$y(x_0 + sh) \simeq u(x_0 + sh) = y_0 + shy'_0 + h^2\sum_{j=0}^{5} \alpha_j(s) f_{c_j}, \tag{2.1.5}$$

where $\beta_j(s) = \int_0^s L_j(r)\,dr, \quad \alpha_j(s) = \int_0^s (s-r)L_j(r)\,dr.$

Local Truncation Error Analysis

Local truncation error (LTE) is a critical concept in the analysis of numerical methods. It quantifies the error introduced in a single step of the method, assuming that all previous values are exact. For a method to be consistent and convergent, the LTE must vanish as the step size $h \to 0$, typically at a rate proportional to a power of h.

In the context of our collocation-based Nystrom scheme, LTE helps us rigorously assess the effect of approximating the second derivative $y''(x)$ by an interpolating polynomial derived from collocation points. Specifically, we expand both the exact and numerical expressions for $y(x_1)$, $y'(x_1)$, and intermediate collocation values using Taylor series. The difference between these

series reveals the leading error term, from which we can derive the method's order of accuracy.

The truncation errors are expressed using the Taylor series expansion in powers of the step size h, where $c_0 = 0,\ c_5 = 1$. These are predefined points for the collocation method. For simplicity and to maintain symmetry in the construction, we fix $c_2 = \frac{1}{2}$. This choice simplifies the structure of the interpolating polynomial and the subsequent integration steps, while still allowing the other collocation points to be optimized to minimize the local truncation error.

Minimizing the leading term in the truncation error allows us to fine-tune the collocation points $c_1,\ c_3,\ c_4$ so that the numerical scheme achieves higher global accuracy. In what follows, we derive the expressions for LTE at key points and determine the conditions under which the error is minimized or canceled.

The variables $c_1,\ c_3,\ c_4,\ s$, and r are defined symbolically to represent unknown collocation points and integration variables.

Local Truncation Error at $y(x_1)$:

At $x_1 = x_0 + h$, the Taylor series for $y(x_1)$ can be expanded as:

$$y(x_1) = y(x_0) + hy'(x_0) + \frac{h^2}{2}y''(x_0) + \cdots + \frac{h^8}{8!}y^{(8)}(x_0) + \mathcal{O}(h^9)$$

$$\Rightarrow \text{LTE}_{y(x_1)} = y(x_1) - \left(y_0 + hy_0' + h^2 \sum_{j=0}^{5} \alpha_j f_{c_j} \right), where$$

$$f_{c_j} = y''(x_{c_j}) = y''(x_0) + c_j h y^{(3)}(x_0) + \frac{(c_j h)^2}{2}y^{(4)}(x_0) + \cdots + \frac{(c_j h)^6}{6!}y^{(8)}(x_0) + \cdots$$

For 8^{th} order as the error term, we obtain

$$\text{LTE}_{y(x_1)} = \frac{h^8}{8!}y^{(8)}(x_0) - h^2 \sum_{j=0}^{5} \alpha_j \frac{(c_j h)^6}{6!}y^{(8)}(x_0)$$

$$\Rightarrow LTE_{y(x_1)} = \frac{h^8}{8!}y^{(8)}(x_0)(1 - 56\sum_{j=0}^{5} \alpha_j {c_j}^6).$$

Now for $y(x_1)$ as $x_1 = x_0 + h,$ since here $s = 1,\ \alpha_j(1) = \int_0^1 (1-r)L_j(r)\,dr$

$$\Rightarrow \quad \sum_{j=0}^{5} \alpha_j {c_j}^6 = \frac{c_1 c_3 c_4}{120} - \frac{c_3}{840} - \frac{c_4}{840} - \frac{c_1}{840} + \frac{2}{105}$$

$$\Rightarrow \mathcal{L}[y(x_1), h] = \frac{(-7c_1c_3c_4 + c_1 + c_3 + c_4 - 1)h^8 y^{(8)}(x_0)}{604800} + \mathcal{O}(h^9).$$

Local Truncation Error at $y(x_{c_3})$:

At $x_{c_3} = x_0 + c_3 h$, the Taylor series for $y(x_{c_3})$ can be expanded as:

$$y(x_{c_3}) = y(x_0) + c_3 h y'(x_0) + \frac{(c_3 h)^2}{2} y''(x_0) + \cdots + \frac{(c_3 h)^8}{8!} y^{(8)}(x_0) + \mathcal{O}(h^9)$$

$$\Rightarrow \text{LTE}_{y(x_{c_3})} = y(x_{c_3}) - \left(y_0 + c_3 h y_0' + h^2 \sum_{j=0}^{5} \alpha_j f_{c_j} \right), \ where$$

$$f_{c_j} = y''(x_{c_j}) = y''(x_0) + c_j h y^{(3)}(x_0) + \frac{(c_j h)^2}{2} y^{(4)}(x_0) + \cdots + \frac{(c_j h)^6}{6!} y^{(8)}(x_0) + \cdots$$

For 8^{th} order as the error term, we obtain

$$\text{LTE}_{y(x_{c_3})} = \frac{(c_3 h)^8}{8!} y^{(8)}(x_0) - h^2 \sum_{j=0}^{5} \alpha_j \frac{(c_j h)^6}{6!} y^{(8)}(x_0)$$

$$\Rightarrow LTE_{y(x_{c_3})} = \frac{(h)^8}{8!} y^{(8)}(x_0)({c_3}^8 - 56 \sum_{j=0}^{5} \alpha_j {c_j}^6).$$

Now for $y(x_{c_3})$ as $x_{c_3} = x_0 + c_3 h$, since here $s = c_3,\ \alpha_j(c_3) = \int_0^{c_3} (c_3 - r) L_j(r)\, dr$

$$\Rightarrow \quad \sum_{j=0}^{5} \alpha_j {c_j}^6 = \begin{aligned} & -\tfrac{c_3^4}{840}\big(14c_1c_3 - 35c_1c_4 + 14c_3c_4 - 21c_1c_3^2 + 8c_1c_3^3 - 21c_3^2c_4 \\ & \quad +8c_3^3c_4 - 7c_3^2 + 12c_3^3 - 20c_3^4 + 42c_1c_3c_4 - 14c_1c_3^2c_4\big) \end{aligned}$$

$$\Rightarrow \mathcal{L}[y(x_{c_3}), h] = \frac{P_{c_3} h^8 y^{(8)}(x_0)}{604800} + \mathcal{O}(h^9), \ where$$

$$\begin{aligned} P_{c_3} = c_3^4\big(c_3\big(& -5c_3^3 + (8c_4 + 12)c_3^2 - (21c_4 + 7)c_3 + 14c_4\big) \\ & + c_1\big(8c_3^3 - (14c_4 + 21)c_3^2 + (42c_4 + 14)c_3 - 35c_4\big)\big). \end{aligned}$$

Local Truncation Error at $y'(x_1)$:

At $x_1 = x_0 + h$, the Taylor series for $y'(x_1)$ can be expanded as:

$$y'(x_1) = y'(x_0) + h y''(x_0) + \frac{h^2}{2} y^{(3)}(x_0) + \cdots + \frac{h^7}{7!} y^{(8)}(x_0) + \mathcal{O}(h^8)$$

$$\Rightarrow \text{LTE}_{y'(x_1)} = y'(x_1) - \left(y_0' + h \sum_{j=0}^{5} \beta_j f_{c_j} \right), \ where$$

$$f_{c_j} = y''(x_{c_j}) = y''(x_0) + c_j h y^{(3)}(x_0) + \frac{(c_j h)^2}{2} y^{(4)}(x_0) + \cdots + \frac{(c_j h)^6}{6!} y^{(8)}(x_0) + \cdots$$

For 7^{th} order as the error term, we obtain

$$\text{LTE}_{y'(x_1)} = \frac{h^7}{7!} y^{(8)}(x_0) - h \sum_{j=0}^{5} \beta_j \frac{(c_j h)^6}{6!} y^{(8)}(x_0)$$

$$\Rightarrow LTE_{y'(x_1)} = \frac{h^7}{7!} y^{(8)}(x_0)(1 - 7 \sum_{j=0}^{5} \beta_j {c_j}^6).$$

Now for $y'(x_1)$ as $x_1 = x_0 + h$, since here $s = 1$, $\beta_j(1) = \int_0^1 L_j(r)\, dr$

$$\Rightarrow \quad \sum_{j=0}^{5} \beta_j {c_j}^6 = \frac{c_1 c_3}{120} - \frac{c_3}{120} - \frac{c_4}{120} - \frac{c_1}{120} + \frac{c_1 c_4}{120} + \frac{c_3 c_4}{120} + \frac{3}{20}$$

$$\Rightarrow \mathcal{L}[y'(x_1), h] = \frac{(-6 - 7c_3(-1 + c_4) + 7c_4 - 7c_1(-1 + c_3 + c_4))\, h^7 y^{(8)}(x_0)}{604800} + \mathcal{O}(h^8).$$

Equating the principal terms in LTE at $y(x_1), y'(x_1), y(x_{c_3})$ to zero,

we have $\begin{cases} -7c_1c_3c_4 + c_1 + c_3 + c_4 - 1 = 0 \\ c_3^4\big(c_3\big(-5c_3^3 + (8c_4 + 12)c_3^2 - (21c_4 + 7)c_3 + 14c_4\big) \\ +c_1\big(8c_3^3 - (14c_4 + 21)c_3^2 + (42c_4 + 14)c_3 - 35c_4\big)\big) = 0 \\ -6 - 7c_3(-1 + c_4) + 7c_4 - 7c_1(-1 + c_3 + c_4) = 0. \end{cases}$

After solving this system, the optimal coefficients derived by this procedure, satisfying $0 < c_1 < \frac{1}{2} < c_3 < c_4 < 1$ are:

$$c_1 = \frac{7 - \sqrt{21}}{14}, \quad c_3 = \frac{7 + \sqrt{21}}{14}, \quad c_4 = \frac{21 + 4\sqrt{21}}{42}.$$

Evaluation at the Step Point

Having constructed the interpolating polynomial for the second derivative and derived the expressions for $u'(x_0 + sh)$ and $u(x_0 + sh)$, we now evaluate these at the step point $s = 1$ to obtain the update formulas for the next numerical values of the solution and its derivative.

This corresponds to computing approximations for $y(x_1)$ and $y'(x_1)$, where $x_1 = x_0 + h$. Substituting the optimal values of the collocation points c_1, c_2, c_3, and applying the integrals involved in the expressions for $\alpha_j(1)$ and $\beta_j(1)$, we obtain closed-form formulas that define the

proposed one-step Nystrom method.These formulas will serve as the core numerical scheme in the implementation and analysis that follows.

Substituting the values of the coefficients and evaluating the formulas in (2.1.4) and (2.1.5) at $s = 1$, we get approximations of $y(x_0 + h)$ and $y'(x_0 + h)$, given as follows:

$$y_1 = y_0 + hy'_0 + \frac{1}{360}h^2 \left(18f_0 + 7\left(\sqrt{21} + 7\right) f_{c_1} + 64f_{c_2} - 7\left(\sqrt{21} - 7\right) f_{c_3}\right), \tag{2.1.6}$$

and

$$hy'_1 = hy'_0 + \frac{1}{180}h^2 \left(9f_0 + 49f_{c_1} + 64f_{c_2} + 49f_{c_3} + 9f_1\right). \tag{2.1.7}$$

Overview of the Derived Scheme

The derivation presented above leads to a high-order one-step Nystrom method tailored specifically for second-order initial value problems. By constructing a seventh-degree interpolating polynomial based on six strategically selected collocation points, and enforcing that the second derivative at these points satisfies the given differential equation, we obtain compact expressions for both $y(x_1)$ and $y'(x_1)$.

The use of collocation allows us to distribute the approximation error in a controlled manner, and the optimized selection of collocation nodes ensures the cancellation of lower-order error terms in the Taylor expansions.

To further assess the method's reliability, we now proceed to a detailed theoretical analysis, focusing on its local truncation error, order of convergence, and global accuracy characteristics.

2.2 Local Trunction Error

After deriving the proposed Nystrom method, it is essential to perform a theoretical analysis to establish its mathematical soundness. A well-designed numerical method must not only produce accurate approximations but also exhibit predictable error behavior, stability, and convergence characteristics.

In this section, we analyze the method from a theoretical standpoint by investigating its local truncation error and order of convergence. These components provide insights into how the method behaves as the step size changes, and whether it delivers the expected accuracy over successive steps.

The analysis also enables us to validate that the carefully chosen collocation points indeed elevate the order of the method, thereby confirming that the optimizations made during derivation are theoretically justified.

In order to implement the proposed Nystrom method in block form, some additional formulas must be considered. To do this, we take the values of $u'(x_0 + sh)$ in (2.1.4) and $u(x_0 + sh)$ in (2.1.5) evaluated at the collocation points obtained for $s = c_1, c_2, c_3, c_4$. In this way, we obtain a total of 10 formulas that form the proposed Nystrom method.

We can rewrite equations (2.1.4) and (2.1.5) as:

$$0 = y'_0 - y'(x_0 + sh) + h \sum_{j=0}^{5} \beta_j(s) f_{c_j}, \tag{2.2.1}$$

and

$$y(x_0 + sh) - y_0 = shy'_0 + h^2 \sum_{j=0}^{5} \alpha_j(s) f_{c_j}. \tag{2.2.2}$$

Interpretation of the Matrix Formulation:

The matrix formulation of the proposed Nystrom method compactly encodes the numerical update rules for both the function values and their derivatives at the collocation points. Each row of the system corresponds to an approximation at a specific point in the interval, either for $y(x)$ or $y'(x)$, and the coefficients capture the contributions of the second derivative evaluations f_{c_j} to the local Taylor expansion.

This representation is not only useful for theoretical analysis, such as truncation error derivation and convergence proofs, but also facilitates efficient numerical implementation, especially in a block setting. Moreover, the structured form of the matrices $\bar{S}$ and $\bar{T}$, built from integrals of Lagrange basis polynomials, highlights the role of collocation in controlling the method's accuracy.

In the following sections, we use this structure to rigorously determine the order of local truncation error and global convergence behavior of the method.

The obtained scheme can be formulated in matrix form as

$$\overline{R}V_o = h\bar{S}V'_o + h^2\overline{T}F_o, \tag{2.2.3}$$

with $\bar{R}, \bar{S}, \bar{T}$ constant matrices containing the coefficients and V_o, V'_o, F_o vector matrices containing variables of the Nystrom scheme, are given by:

$$\bar{R}\mathbf{V}_0 = \begin{bmatrix} -1 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} y_0 \\ y_{c_1} \\ y_{c_2} \\ y_{c_3} \\ y_{c_4} \\ y_1 \end{bmatrix}; \quad h\bar{S}\mathbf{V}_0' = h \begin{bmatrix} c_1 & 0 & 0 & 0 & 0 & 0 \\ c_2 & 0 & 0 & 0 & 0 & 0 \\ c_3 & 0 & 0 & 0 & 0 & 0 \\ c_4 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} y_0' \\ y_{c_1}' \\ y_{c_2}' \\ y_{c_3}' \\ y_{c_4}' \\ y_1' \end{bmatrix};$$

$$h^2\overline{T}F_o = h^2 \begin{bmatrix} \alpha_0(c_1) & \alpha_1(c_1) & \alpha_2(c_1) & \alpha_3(c_1) & \alpha_4(c_1) & \alpha_5(c_1) \\ \alpha_0(c_2) & \alpha_1(c_2) & \alpha_2(c_2) & \alpha_3(c_2) & \alpha_4(c_2) & \alpha_5(c_2) \\ \alpha_0(c_3) & \alpha_1(c_3) & \alpha_2(c_3) & \alpha_3(c_3) & \alpha_4(c_3) & \alpha_5(c_3) \\ \alpha_0(c_4) & \alpha_1(c_4) & \alpha_2(c_4) & \alpha_3(c_4) & \alpha_4(c_4) & \alpha_5(c_4) \\ \alpha_0(1) & \alpha_1(1) & \alpha_2(1) & \alpha_3(1) & \alpha_4(1) & \alpha_5(1) \\ \beta_0(c_1) & \beta_1(c_1) & \beta_2(c_1) & \beta_3(c_1) & \beta_4(c_1) & \beta_5(c_1) \\ \beta_0(c_2) & \beta_1(c_2) & \beta_2(c_2) & \beta_3(c_2) & \beta_4(c_2) & \beta_5(c_2) \\ \beta_0(c_3) & \beta_1(c_3) & \beta_2(c_3) & \beta_3(c_3) & \beta_4(c_3) & \beta_5(c_3) \\ \beta_0(c_4) & \beta_1(c_4) & \beta_2(c_4) & \beta_3(c_4) & \beta_4(c_4) & \beta_5(c_4) \\ \beta_0(1) & \beta_1(1) & \beta_2(1) & \beta_3(1) & \beta_4(1) & \beta_5(1) \end{bmatrix} \begin{bmatrix} f_0 \\ f_{c_1} \\ f_{c_2} \\ f_{c_3} \\ f_{c_4} \\ f_1 \end{bmatrix},$$

here, $\alpha_j(s) = \int_0^s (s-r)L_j(r)\,dr$, $\beta_j(s) = \int_0^s L_j(r)\,dr$,

and, $c_1 = \frac{7-\sqrt{21}}{14}$, $c_2 = \frac{1}{2}$, $c_3 = \frac{7+\sqrt{21}}{14}$, $c_4 = \frac{21+4\sqrt{21}}{42}$.

The matrix $\bar{T}$, composed of entries $\alpha_j(s)$ and $\beta_j(s)$, represents the integral contributions of the Lagrange basis functions and their weighted moments over the interval $[0, s]$. These integrals play a central role in reconstructing the numerical solution and its derivative from the second derivative evaluations f_{c_j} at collocation points.

Furthermore, the specific structure of the $\bar{T}$ matrix is designed to align with the symmetry and polynomial degree of the collocation system, reinforcing the global order of the method.

Now, by evaluating the expressions for $\alpha_j(s)$ and $\beta_j(s)$ at $s = c_1, c_2, c_3, c_4, 1$ using MATLAB Codes (4.4) and (4.5), we obtain the following coefficient matrix $\bar{T}$:

$$\bar{T} = \begin{pmatrix}
\frac{1111-171\sqrt{21}}{41160} & \frac{55}{6174} & \frac{2(419-98\sqrt{21})}{15435} & \frac{1709-343\sqrt{21}}{17640} & \frac{-18}{1715} & \frac{12\sqrt{21}+41}{20580} \\
\frac{49}{1920} & \frac{7(8\sqrt{21}+35)}{5760} & \frac{1}{72} & \frac{-7(8\sqrt{21}-35)}{5760} & 0 & \frac{1}{1920} \\
\frac{171\sqrt{21}+919}{41160} & \frac{2401\sqrt{21}+11003}{123480} & \frac{2(98\sqrt{21}+449)}{15435} & \frac{1}{882} & \frac{18}{1715} & \frac{-12\sqrt{21}-55}{20580} \\
\frac{88612\sqrt{21}+426493}{17781120} & \frac{49\sqrt{\frac{7}{3}}}{720} + \frac{5545181}{53343360} & \frac{129361}{1666980} + \frac{16}{45\sqrt{21}} & \frac{7\sqrt{\frac{7}{3}}}{720} + \frac{112019}{762480} & \frac{25}{24696} & \frac{-788\sqrt{21}-3607}{3556224} \\
\frac{1}{20} & \frac{7}{360}(\sqrt{21}+7) & \frac{8}{45} & \frac{-7(\sqrt{21}-7)}{360} & 0 & 0 \\
\frac{41\sqrt{21}+189}{5880} & \frac{49}{360} - \frac{23}{840\sqrt{21}} & \frac{8}{45} - \frac{106}{105\sqrt{21}} & \frac{343-29\sqrt{21}}{2520} & -\frac{1}{35}(6\sqrt{\frac{3}{7}}) & \frac{1}{56} + \frac{41}{280\sqrt{21}} \\
\frac{1}{960}(81-8\sqrt{21}) & \frac{95\sqrt{21}+392}{2880} & \frac{\sqrt{\frac{7}{3}}}{48} + \frac{8}{45} & \frac{-7(25\sqrt{21}-56)}{2880} & \frac{3\sqrt{21}}{80} & \frac{-8\sqrt{21}-33}{960} \\
\frac{23\sqrt{21}+189}{5880} & \frac{49}{360} + \frac{523}{840\sqrt{21}} & \frac{8}{45} + \frac{86}{105\sqrt{21}} & \frac{7}{360}(\sqrt{21}+7) & -\frac{1}{35}(6\sqrt{\frac{3}{7}}) & \frac{1}{56} + \frac{23}{280\sqrt{21}} \\
\frac{3368\sqrt{21}+48027}{1270080} & \frac{49}{360} + \frac{113269}{181440\sqrt{21}} & \frac{8}{45} + \frac{36871}{45360\sqrt{21}} & \frac{49}{360} + \frac{16717}{25920\sqrt{21}} & \frac{-65}{336\sqrt{21}} & \frac{737}{60480} + \frac{421}{7560\sqrt{21}} \\
\frac{1}{20} & \frac{49}{180} & \frac{16}{45} & \frac{49}{180} & 0 & \frac{1}{20}
\end{pmatrix}$$

Each row of the matrices $\bar{R}$, $\bar{S}$, and $\bar{T}$ corresponds to a specific collocation condition. Specifically, the first row represents the equation at y_{c_1}, the second at y_{c_2}, the third at y_{c_3}, the fourth at y_{c_4}, the fifth at y_1, and the remaining rows correspond to derivative evaluations at y'_{c_1}, y'_{c_2}, y'_{c_3}, y'_{c_4}, and y'_1, respectively. This ordering ensures that the collocation system accounts for both solution and derivative values in a structured and consistent manner.

Order of Local Truncation Error:

An important aspect of evaluating a numerical method is to determine its local truncation error (LTE), which quantifies the error introduced in a single integration step assuming exact knowledge of previous values. The LTE provides direct insight into the consistency and theoretical convergence of the method.

For collocation-based Nystrom schemes, this error is typically expressed in terms of Taylor series expansions of the exact solution and its derivatives. By comparing these expansions with the numerical formulation, we can isolate the leading-order term in the error expression.

To formalize this process, we define an operator that reflects the algebraic structure of the method and captures how the interpolated solution deviates from the true solution at each step. The order of this deviation as a function of the step size h indicates the theoretical accuracy of the scheme.

We define the operator Γ related to the proposed Nystrom method in (2.2.3), assuming that $y(x)$ has enough derivatives, as

$$\Gamma[y(x);h] = \sum_{j\in I} \left[\theta_j y(x+jh) - h\Theta_j y'(x+jh) - h^2\vartheta_j y''(x+jh)\right], \tag{2.2.4}$$

where θ_j, Θ_j, and ϑ_j are, respectively, vector columns of $\bar{R}$, $\bar{S}$, and $\bar{T}$, and $I = \{0, c_1, c_2, c_3, c_4, 1\}$.

Expanding $y(x_0+jh)$, $y'(x_0+jh)$, and $y''(x_0+jh)$ in Taylor series about x_0, we have

$$y(x_0+jh) = y(x_0) + jh\,y'(x_0) + \frac{j^2h^2}{2!}y''(x_0) + \cdots + \frac{j^qh^q}{q!}y^{(q)}(x_0) + \cdots$$

$$y'(x_0+jh) = y'(x_0) + jh\,y''(x_0) + \frac{j^2h^2}{2!}y'''(x_0) + \cdots + \frac{j^{q-1}h^{q-1}}{(q-1)!}y^{(q)}(x_0) + \cdots$$

$$y''(x_0+jh) = y''(x_0) + jh\,y'''(x_0) + \frac{j^2h^2}{2!}y^{(4)}(x_0) + \cdots + \frac{j^{q-2}h^{q-2}}{(q-2)!}y^{(q)}(x_0) + \cdots$$

Thus by putting those values in (2.2.4), we get

$$\Gamma[y(x);h] = C_0y(x) + C_1hy'(x) + C_2h^2y''(x) + \cdots + C_qh^qy^{(q)}(x) + \ldots,$$

where

$$C_q = \frac{1}{q!}\left[\sum_{j\in I} j^q\theta_j - q\sum_{j\in I} j^{q-1}\Theta_j - q(q-1)\sum_{j\in I} j^{q-2}\vartheta_j\right], \tag{2.2.5}$$

and $q = 0, 1, 2, 3, 4, \ldots$.

The order of the LTE is determined by the first non-null vector C_q. In other words, C_q corresponds to the lowest-order derivative term whose coefficient does not vanish in the Taylor expansion. The smallest such q for which $C_q \neq 0$ dictates the leading term in the local error and, consequently, determines the order of the method to be q. Thus, we get following results as follows:

Formula:	y_{c_1}	y_{c_2}	y_{c_3}	y_{c_4}	y_1	y'_{c_1}	y'_{c_2}	y'_{c_3}	y'_{c_4}	y'_1
Order (LTE):	8	8	9	8	9	7	7	7	7	9

For solving (2.2.5), one can go through MATLAB CODEs (4.6), (4.7).

Also, we will find LTE for the above Formulas regarding their orders.

Local Truncation Error at $y(x_{c_j})$:

At $x_{c_j} = x_0 + c_j h$: The Taylor series for $y(x_{c_j})$ is expanded as:

$$y(x_{c_j}) = y(x_0) + c_j h y'(x_0) + \frac{(c_j h)^2}{2} y''(x_0) + \cdots + \frac{(c_j h)^8}{8!} y^{(8)}(x_0) + \mathcal{O}(h^9)$$

$$\Rightarrow \text{LTE}_{y(x_{c_j})} = y(x_{c_j}) - \left(y_0 + c_j h y_0' + h^2 \sum_{j=0}^{5} \alpha_j f_{c_j} \right), \ where$$

$$f_{c_j} = y''(x_{c_j}) = y''(x_0) + c_j h y^{(3)}(x_0) + \frac{(c_j h)^2}{2} y^{(4)}(x_0) + \cdots + \frac{(c_j h)^6}{6!} y^{(8)}(x_0) + \cdots$$

For 8^{th} order as the error term, we obtain

$$\text{LTE}_{y(x_{c_j})} = \frac{(c_j h)^8}{8!} y^{(8)}(x_0) - h^2 \sum_{j=0}^{5} \alpha_j \frac{(c_j h)^6}{6!} y^{(8)}(x_0)$$

$$\Rightarrow LTE_{y(x_{c_j})} = \frac{(h)^8}{8!} y^{(8)}(x_0)({c_j}^8 - 56 \sum_{j=0}^{5} \alpha_j {c_j}^6); \quad (for \ y_{c_1}, \ y_{c_2}, \ y_{c_4}).$$

For 9^{th} order as the error term, we obtain

$$\text{LTE}_{y(x_{c_j})} = \frac{(c_j h)^9}{9!} y^{(9)}(x_0) - h^2 \sum_{j=0}^{5} \alpha_j \frac{(c_j h)^7}{7!} y^{(9)}(x_0)$$

$$\Rightarrow LTE_{y(x_{c_j})} = \frac{(h)^9}{9!} y^{(9)}(x_0)({c_j}^9 - 72 \sum_{j=0}^{5} \alpha_j {c_j}^7); \quad (for \ y_{c_3}, \ y_1).$$

Formula	**Local truncation errors**
y_{c_1}	$-\frac{h^8 y^{(8)}(x_0)}{36303120} + \mathcal{O}(h^9)$
y_{c_2}	$-\frac{h^8 y^{(8)}(x_0)}{30965760} + \mathcal{O}(h^9)$
y_{c_3}	$\frac{(523\sqrt{21}+2401) h^9 y^{(9)}(x_0)}{853849382400} + \mathcal{O}(h^{10})$
y_{c_4}	$-\frac{125 h^8 y^{(8)}(x_0)}{1204450394112} + \mathcal{O}(h^9)$
y_1	$\frac{h^9 y^{(9)}(x_0)}{177811200} + \mathcal{O}(h^{10})$

Local Truncation Error at $y'(x_{c_j})$:

At $x_{c_j} = x_0 + c_j h$: The Taylor series for $y'(x_{c_j})$ is expanded as:

$$y'(x_{c_j}) = y'(x_0) + c_j h y''(x_0) + \frac{(c_j h)^2}{2!} y^{(3)}(x_0) + \cdots + \frac{(c_j h)^7}{7!} y^{(8)}(x_0) + \mathcal{O}(h^8)$$

$$\Rightarrow \text{LTE}_{y'(x_{c_j})} = y'(x_{c_j}) - \left(y'_0 + h \sum_{j=0}^{5} \beta_j f_{c_j} \right), \ where$$

$$f_{c_j} = y''(x_{c_j}) = y''(x_0) + c_j h y^{(3)}(x_0) + \frac{(c_j h)^2}{2} y^{(4)}(x_0) + \cdots + \frac{(c_j h)^6}{6!} y^{(8)}(x_0) + \cdots$$

For 7^{th} order as the error term, we obtain

$$LTE_{y'(x_{c_j})} = \frac{(c_j h)^7}{7!} y^{(8)}(x_0) - h \sum_{j=0}^{5} \beta_j \frac{(c_j h)^6}{6!} y^{(8)}(x_0)$$

$$\Rightarrow LTE_{y'(x_{c_j})} = \frac{(c_j h)^7}{7!} y^{(8)}(x_0)(1 - 7 \sum_{j=0}^{5} \beta_j {c_j}^6); \quad (for\ y'_{c_1},\ y'_{c_2},\ y'_{c_3},\ y'_{c_4}).$$

For 9^{th} order as the error term, we obtain

$$\text{LTE}_{y'(x_1)} = \frac{h^9}{9!} y^{(10)}(x_0) - h \sum_{j=0}^{5} \beta_j \frac{(h)^8}{8!} y^{(10)}(x_0)$$

$$\Rightarrow LTE_{y'(x_1)} = \frac{h^9}{9!} y^{(10)}(x_0)(1 - 9 \sum_{j=0}^{5} \beta_j {c_j}^8).$$

Formula	**Local truncation errors**
y'_{c_1}	$-\frac{11h^7 y^{(8)}(x_0)}{8297856\sqrt{21}} + \mathcal{O}(h^8)$
y'_{c_2}	$\frac{h^7 y^{(8)}(x_0)}{967680\sqrt{21}} + \mathcal{O}(h^8)$
y'_{c_3}	$-\frac{h^7 y^{(8)}(x_0)}{41489280\sqrt{21}} + \mathcal{O}(h^8)$
y'_{c_4}	$\frac{25h^7 y^{(8)}(x_0)}{1792336896\sqrt{21}} + \mathcal{O}(h^8)$
y'_1	$-\frac{h^9 y^{(10)}(x_0)}{1422489600} + \mathcal{O}(h^{10})$

The local truncation error analysis confirms that the proposed Nystrom method achieves a high order of accuracy at both the collocation points and the endpoint $x_1 = x_0 + h$. By carefully choosing the collocation nodes and evaluating the interpolated polynomial integrals, we ensured the cancellation of lower-order error terms in the Taylor expansions.

The method yields an eighth-order approximation for the solution at x_1, and seventh-order approximations at the intermediate points, while the derivative approximations also achieve seventh-order or higher depending on the evaluation point.

These results demonstrate the theoretical consistency and precision of the method, validating the success of the optimization strategy used during derivation. In the next section, we transition

from local behavior to the global order of accuracy of the entire integration scheme.

Order of Accuracy:

While the local truncation error describes the error introduced at a single step under the assumption of exact past values, the global order of accuracy measures the cumulative effect of these errors across all steps of the integration process. It reflects the rate at which the global error decreases as the step size h tends to zero.

For collocation-based methods like the one proposed here, the global accuracy depends not only on the local truncation error but also on the structure of the collocation system and the smoothness of the underlying solution. In particular, the theory established by van der Houwen et al.[3] provides conditions under which a collocation-based Nystrom method achieves higher global step point order.

In the present case, the method satisfies the required conditions for increased global accuracy due to the special selection of collocation points. As a result, we obtain a global step point order of $p = 8$, ensuring that the solution and its derivative converge rapidly to the exact solution as the mesh is refined. Here, we note that the resulting scheme is a Nystrom method based on direct collocation. According to van der Houwen et al.[3], if

$$y(x_1) - y_1 = \mathcal{O}\left(h^{p_1+1}\right), \quad y'(x_1) - y'_1 = \mathcal{O}\left(h^{p_2+1}\right),$$

then the order of accuracy is defined as $p = \min(p_1, p_2)$.

By van der Houwen et al.[3, *theorem* 3.2], we know that the method has global step point order of at least $p = s = 6$, and locally, the order of the y-component is 7, and the order of the y' component is one lower. We have increased the global step point order with the special choice of the collocation points. Since van der Houwen et al.[3, *theorem* 3.4] is satisfied with

$$p = s + q = 6 + 2 = 8,$$

then the proposed Nystom method has global step point order $p = 8$.

Formula:	y_{c_1}	y_{c_2}	y_{c_3}	y_{c_4}	y_1	y'_{c_1}	y'_{c_2}	y'_{c_3}	y'_{c_4}	y'_1
Order of accuracy:	7	7	7	7	8	6	6	6	6	8

The results in the table clearly demonstrate that the proposed method achieves high-order accuracy at all evaluation points, with a global step point order of 8 for the solution and 7 or higher for its derivative. These orders significantly exceed those of classical Nystrom methods, particularly at internal collocation points.

This confirms that the optimization strategy and collocation design have successfully enhanced both the precision and efficiency of the numerical scheme, making it well-suited for solving a wide range of second-order initial value problems with high fidelity.

2.3 Overview of the Developed Scheme

In this chapter, we developed a high-order collocation-based Nystrom scheme for solving second-order initial value problems. Beginning with the derivation based on Taylor expansion and collocation, we constructed a one-step scheme capable of directly approximating both the solution and its derivative without reducing the problem to a system of first-order equations.

Through careful selection of collocation points and analytical integration of Lagrange basis functions, we designed the method to minimize local truncation errors and achieve high global step point accuracy. Theoretical analysis confirmed that the method attains eighth-order accuracy at the endpoint and seventh-order accuracy at intermediate points, for both the solution and its derivative.

These findings demonstrate that the proposed method is not only mathematically robust but also well-suited for practical implementation. In the next chapter, we validate the method through a series of numerical experiments on both linear and nonlinear second-order problems, further illustrating its accuracy and efficiency.

Chapter 3

Numerical Experiments

The derivation and theoretical analysis of the proposed Nystrom method have established its high-order accuracy and solid mathematical foundation. Through Taylor expansion and careful selection of collocation points, we have ensured the minimization of local truncation errors and confirmed the method's global step point order. The practical performance of any numerical method is tested through some numerical examples. It is crucial to validate the scheme through implementation on a variety of test problems, including both linear and nonlinear second-order initial value problems.

In this chapter, we present a series of numerical experiments designed to demonstrate the accuracy, and robustness of the proposed method. These results confirm the theoretical claims and highlight the method's potential for solving real-world differential equations.

Steps to be followed for numerical implimentation:

Consider the initial value problem (IVP):

$$y''(x) = f(x, y, y'), \quad y(x_0) = y_0, \quad y'(x_0) = y'_0 \tag{3.1.1}$$

We define:

$$f_0 = y''(x_0) = f(x_0, y_0, y'_0)$$

Differentiating Eq.(3.1.1) up to the 8^{th} order, we get

$$y^{(3)}(x) = f(x, y, y', y'')$$

$$y^{(4)}(x) = f(x, y, y', y'', y^{(3)})$$

$$\vdots$$

$$y^{(8)}(x) = f(x, y, y', y'', \ldots, y^{(7)})$$

Hence, evaluating at x_0, we obtain

$$y^{(3)}(x_0) = f(x_0, y_0, y_0', y_0'')$$

$$y^{(4)}(x_0) = f(x_0, y_0, y_0', y_0'', y_0^{(3)})$$

$$\vdots$$

$$y^{(8)}(x_0) = f(x_0, y_0, y_0', y_0'', \ldots, y_0^{(7)})$$

To evaluate $f(x_{c_j})$, where $x_{c_j} = x_0 + c_j h$. We consider a Taylor series expansion about x_0:

$$f_{c_j} = y''(x_{c_j}) = y''(x_0 + c_j h)$$

Expanding $y''(x)$ in Taylor series:

$$f_{c_1} = y''(x_0) + c_1 h y^{(3)}(x_0) + \cdots + \frac{(c_1 h)^6}{6!} y^{(8)}(x_0)$$

$$f_{c_2} = y''(x_0) + c_2 h y^{(3)}(x_0) + \cdots + \frac{(c_2 h)^6}{6!} y^{(8)}(x_0)$$

$$f_{c_3} = y''(x_0) + c_3 h y^{(3)}(x_0) + \cdots + \frac{(c_3 h)^6}{6!} y^{(8)}(x_0)$$

$$f_1 = y''(x_1) = y''(x_0 + h) = y''(x_0) + h y^{(3)}(x_0) + \cdots + \frac{h^6}{6!} y^{(8)}(x_0)$$

Thus, using the values $f_0,\ f_{c_1},\ f_{c_2},\ f_{c_3},\ f_1$, we get the expressions:

$$y(x_1) = y_0 + h y_0' + \frac{h^2}{360}\left[186 f_0 + 7\left(\sqrt{21} + 7\right) f_{c_2} + 64 f_{c_3} - 7\left(\sqrt{21} - 7\right) f_{c_1}\right],$$

and

$$y'(x_1) = y_0' + \frac{h}{180}\left[96 f_0 + 49 f_{c_1} + 64 f_{c_2} + 49 f_{c_3} + 9 f_1\right].$$

3.1 Numerical Results (2nd-order linear IVPs)

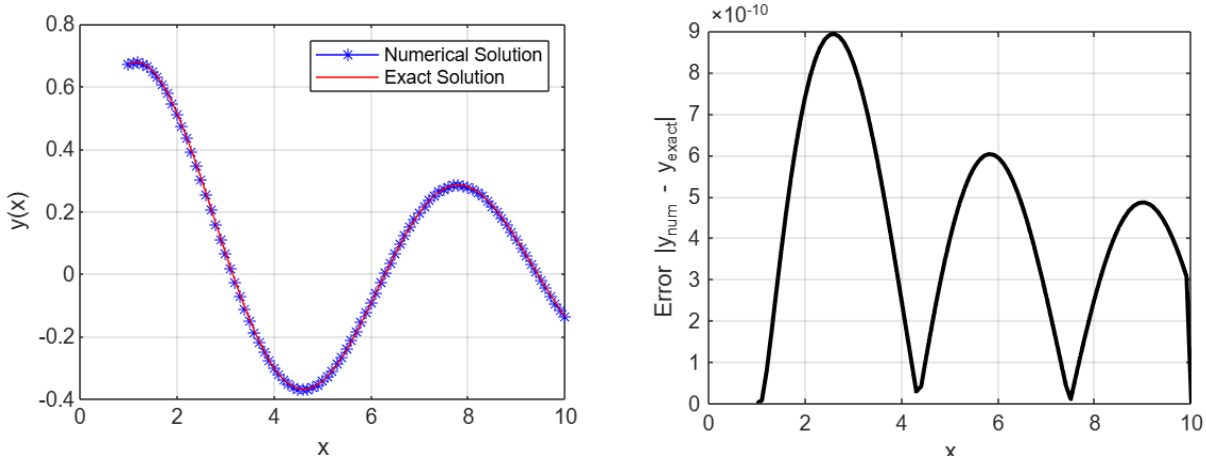
Example 3.1. Consider the following bessel problem:

$$x^2y'' + xy' + (x^2 - 0.25)y = 0, \quad 1 \le x \le 10,$$
$$y(1) = \sqrt{\frac{2}{\pi}}\sin(1), \quad y'(1) = \frac{(2\cos(1) - \sin(1))}{\sqrt{(2\pi)}},$$

whose exact solution is

$$y(x) = \sqrt{\frac{2}{\pi x}}\sin(x).$$

Sr.No.	**Step Size** (h)	**MAE**
1	$h = 1/10$	8.934196×10^{-10}
2	$h = 1/20$	6.052880×10^{-12}
3	$h = 1/40$	4.374279×10^{-14}
4	$h = 1/80$	9.436896×10^{-16}



(a) Numerical Solution vs Exact Solution

(b) Error Plot

Figure 3.1: Comparison of the Numerical Solution with the Exact Solution for problem 3.1.

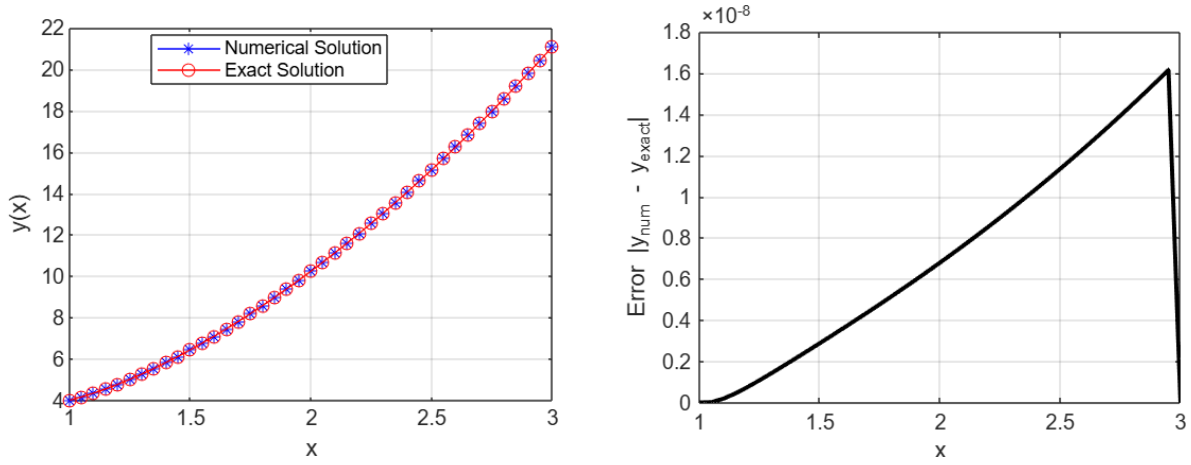
Example 3.2. Consider the 2nd order initial value problem:

$$x^2y'' + xy' - 4y = -3x, \quad 1 \leq x \leq 3,$$
$$y(1) = 4, \quad y'(1) = 3,$$

whose exact solution is

$$y(x) = 2x^2 + x + x^{-2}.$$

Sr.No.	**Step Size** (h)	**MAE**
1	$h = 1/10$	2.314112×10^{-6}
2	$h = 1/20$	1.615876×10^{-8}
3	$h = 1/40$	1.182769×10^{-10}
4	$h = 1/80$	8.846257×10^{-13}



(a) Numerical Solution vs Exact Solution

(b) Error Plot

Figure 3.2: Comparison of the Numerical Solution with the Exact Solution for problem 3.2.

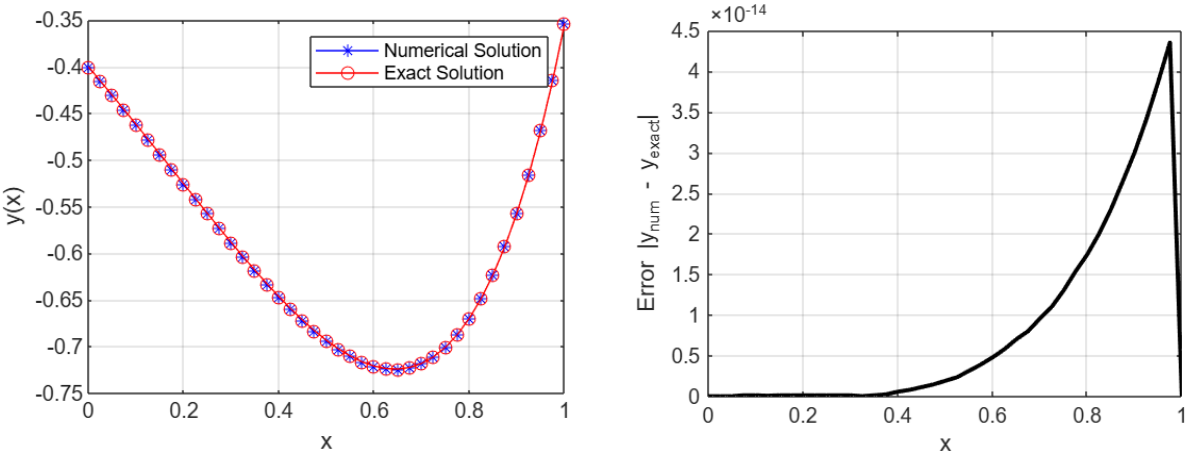
Example 3.3. Consider the 2nd order initial value problem:

$$y'' - 2y' + 2y = e^{2x}sin(x), \quad 0 \leq x \leq 1,$$
$$y(0) = -0.4, \quad y'(1) = -0.6,$$

whose exact solution is

$$y(x) = \frac{1}{5}e^{2x}(sin(x) - 2cos(x)).$$

Sr.No.	**Step Size** (h)	**MAE**
1	$h = 1/10$	3.421757×10^{-10}
2	$h = 1/20$	4.383771×10^{-12}
3	$h = 1/40$	4.357625×10^{-14}
4	$h = 1/80$	3.330669×10^{-16}



(a) Numerical Solution vs Exact Solution

(b) Error Plot

Figure 3.3: Comparison of the Numerical Solution with the Exact Solution for problem 3.3.

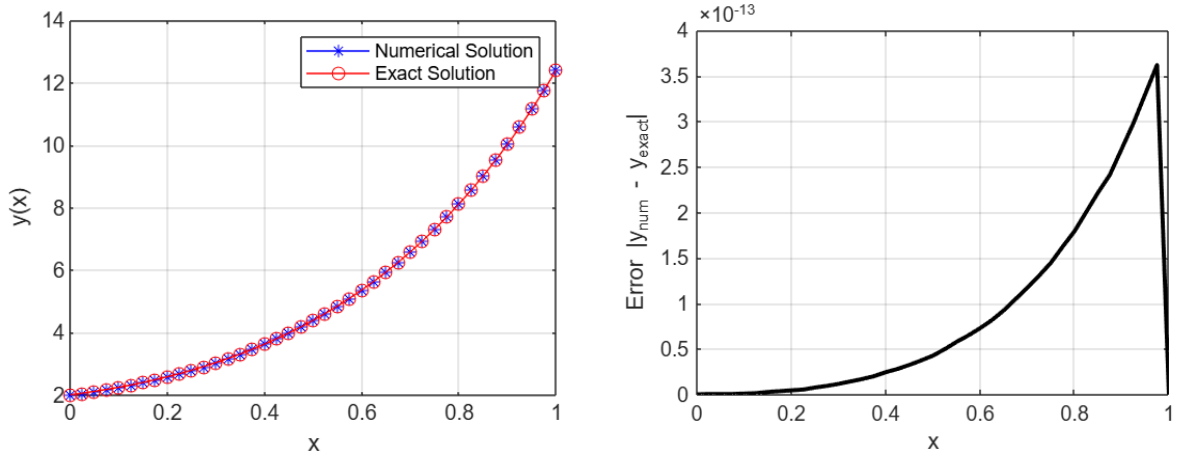
Example 3.4. Consider the 2nd order initial value problem:

$$y'' - 3y' + 2y = 6e^{-x}, \quad 0 \leq x \leq 1,$$
$$y(0) = 2, \quad y'(0) = 2,$$

whose exact solution is

$$y(x) = 2e^{2x} - e^{x} + e^{-x}.$$

Sr.No.	**Step Size** (h)	**MAE**
1	$h = 1/10$	3.600659×10^{-9}
2	$h = 1/20$	3.915801×10^{-11}
3	$h = 1/40$	3.623768×10^{-13}
4	$h = 1/80$	3.552714×10^{-15}



(a) Numerical Solution vs Exact Solution

(b) Error Plot

Figure 3.4: Comparison of the Numerical Solution with the Exact Solution for problem 3.4.

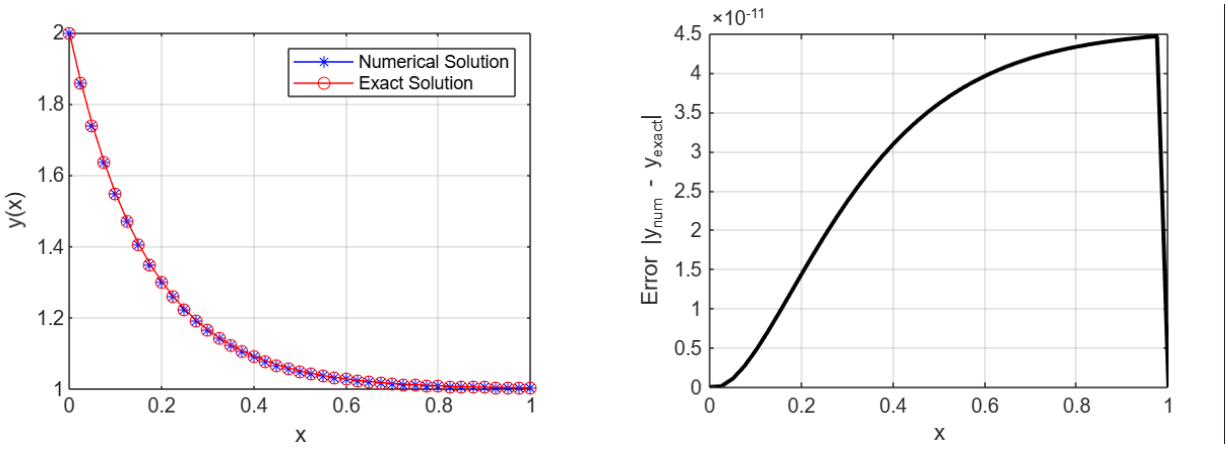
Example 3.5. Consider the 2nd order initial value problem:

$$y'' + 6y' = 0, \quad 0 \leq x \leq 1,$$
$$y(0) = 2, \quad y'(0) = -6,$$

whose exact solution is

$$y(x) = 1 + e^{-6x}.$$

Sr.No.	**Step Size** (h)	**MAE**
1	$h = 1/10$	8.901908×10^{-7}
2	$h = 1/20$	6.122215×10^{-9}
3	$h = 1/40$	4.468625×10^{-11}
4	$h = 1/80$	3.368417×10^{-13}



(a) Numerical Solution vs Exact Solution

(b) Error Plot

Figure 3.5: Comparison of the Numerical Solution with the Exact Solution for problem 3.5.

3.2 Numerical results (2nd-order non-linear IVPs)

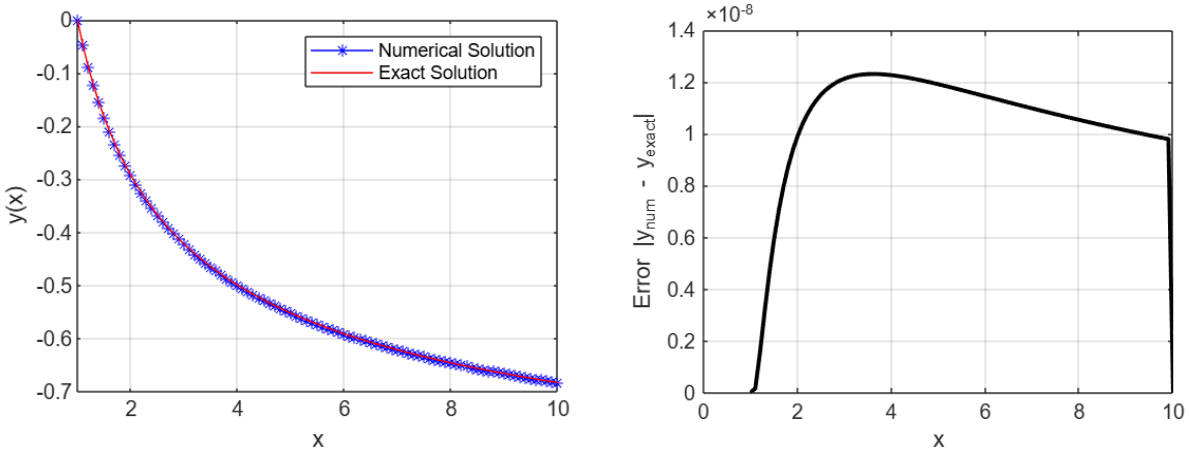
Example 3.6. Consider the non-linear homogeneous problem:

$$(y+1)y'' - 3(y')^2 = 0, \quad 1 \le x \le 10,$$
$$y(1) = 0, \quad y'(1) = -\frac{1}{2},$$

whose exact solution is

$$y(x) = \frac{1}{\sqrt{x}} - 1.$$

Sr.No.	**Step Size** (h)	**MAE**
1	$h = 1/10$	1.231759×10^{-8}
2	$h = 1/20$	8.034445×10^{-11}
3	$h = 1/40$	5.711542×10^{-13}
4	$h = 1/80$	4.163336×10^{-15}



(a) Numerical Solution vs Exact Solution

(b) Error Plot

Figure 3.6: Comparison of the Numerical Solution with the Exact Solution for problem 3.6.

Use of the Double Mesh Principle

In the previous examples (Examples 3.1 to 3.6), the accuracy of the proposed method was assessed by computing the maximum absolute error (MAE) between the numerical solution and the known exact solution. However, for some of the subsequent problems, exact solutions are not available in closed form.

To estimate the numerical error in such cases, we apply the *double mesh principle*, a standard approach for assessing convergence in the absence of exact solutions. This method involves solving the same problem using two different step sizes, typically h and $h/2$, and using their difference as an approximation of the error.

Let y_h and $y_{h/2}$ denote the numerical solutions computed on coarse and fine meshes, respectively. Then, the estimated error at a point x_i is given by:

$$\varepsilon_i \approx \left| y_{h/2}(x_i) - y_h(x_i) \right|,$$

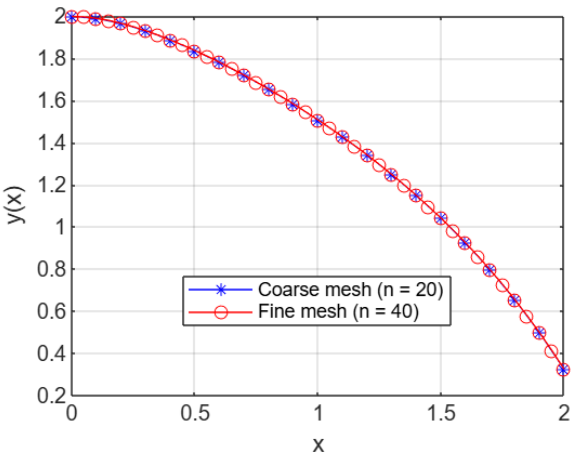
where both approximations are evaluated at common mesh points. This approach allows us to assess the convergence behavior of the method even when the exact solution is not available.

Example 3.7. Consider the non-linear homogeneous problem:

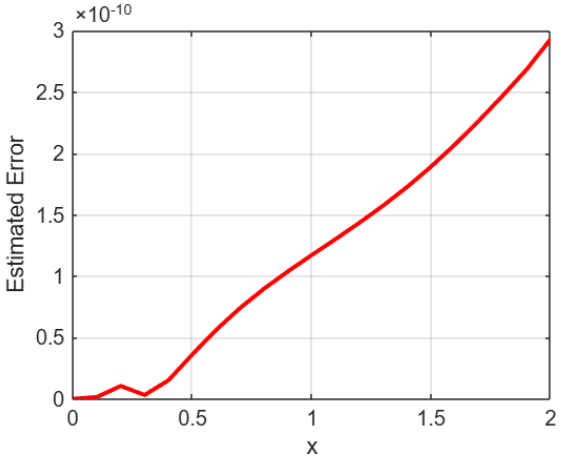
$$y'' = v(1-y^2)y' - y, \quad 0 \le x \le 2,$$
$$y(0) = 2, \quad y'(0) = 0.$$

Taking $v = 1$, we have following results:

Sr.No.	**Step Size** (h)	**MAE**
1	$h = 1/10$	2.932433×10^{-10}
2	$h = 1/20$	2.605173×10^{-12}
3	$h = 1/40$	2.135431×10^{-14}
4	$h = 1/80$	2.189849×10^{-16}



(a) Coarse mesh vs Fine mesh Solution



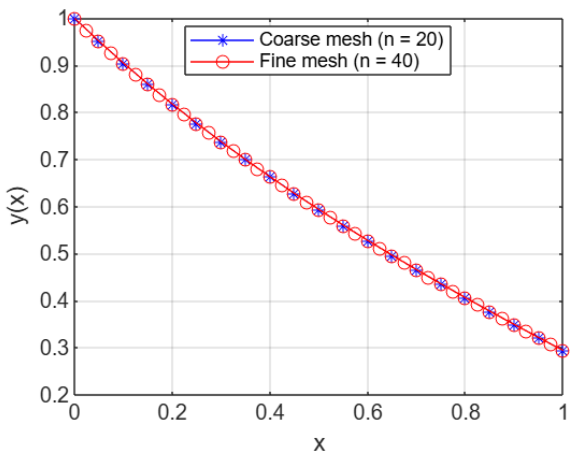
(b) Estimated Error via Double Mesh Principle

Figure 3.7: Numerical solution and error plot for problem 3.7.

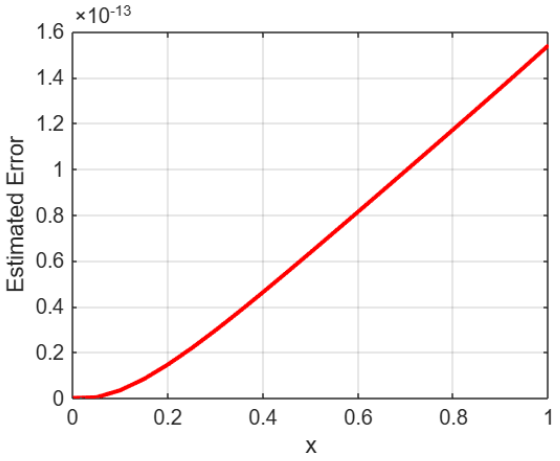
Example 3.8. Consider the non-linear homogeneous problem:

$$y'' + yy' = 0, \quad 0 \leq x \leq 1,$$
$$y(0) = 1, \quad y'(0) = -1.$$

Sr.No.	**Step Size (h)**	**MAE**
1	$h = 1/10$	2.153059×10^{-11}
2	$h = 1/20$	1.541872×10^{-13}
3	$h = 1/40$	1.147813×10^{-15}
4	$h = 1/80$	4.370957×10^{-18}



(a) Coarse mesh vs Fine mesh Solution



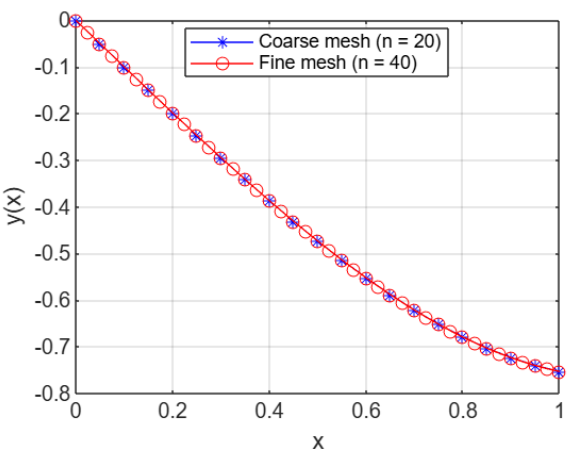
(b) Estimated Error via Double Mesh Principle

Figure 3.8: Numerical solution and error plot for problem 3.8.

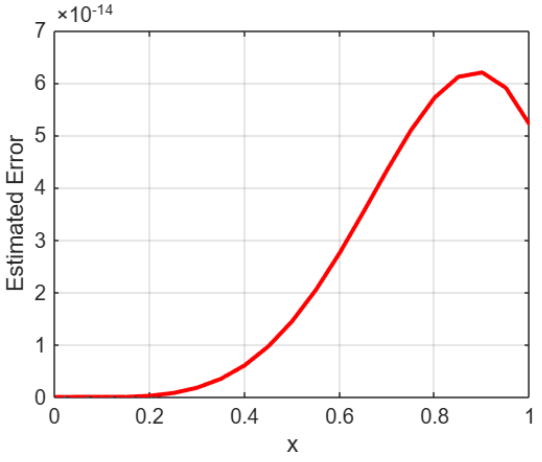
Example 3.9. Consider the non-linear homogeneous problem:

$$y'' + ye^{-y} = 0, \quad 0 \leq x \leq 1,$$
$$y(0) = 0, \quad y'(0) = -1.$$

Sr.No.	**Step Size (h)**	**MAE**
1	$h = 1/10$	7.849119×10^{-12}
2	$h = 1/20$	6.205010×10^{-14}
3	$h = 1/40$	4.843020×10^{-16}
4	$h = 1/80$	6.119340×10^{-18}



(a) Coarse mesh vs Fine mesh Solution



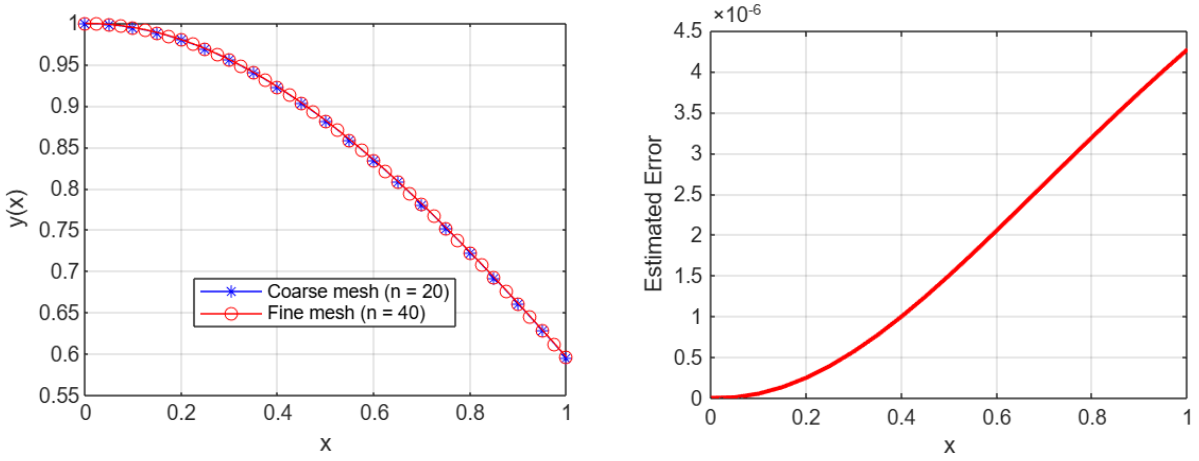
(b) Estimated Error via Double Mesh Principle

Figure 3.9: Numerical solution and error plot for problem 3.9.

Example 3.10. Consider the non-linear homogeneous problem:

$$y'' + y^3 = 0, \quad 0 \leq x \leq 1,$$
$$y(0) = 1, \quad y'(0) = 0.$$

Sr.No.	Step Size (h)	MAE
1	$h = 1/10$	1.693685×10^{-5}
2	$h = 1/20$	4.273954×10^{-6}
3	$h = 1/40$	1.073501×10^{-6}
4	$h = 1/80$	2.689981×10^{-7}



(a) Coarse mesh vs Fine mesh Solution (b) Estimated Error via Double Mesh Principle

Figure 3.10: Numerical solution and error plot for problem 3.10.

Summary of Numerical Results

The numerical experiments presented in this chapter validate the theoretical performance of the proposed collocation-based Nystrom scheme. Both linear and nonlinear second-order IVPs were solved with high accuracy, confirming the predicted local and global error orders. The method demonstrated excellent convergence behavior, with mean absolute errors decreasing rapidly as the step size was refined. These results support the method's applicability to a broad class of second-order problems.

Chapter 4

Conclusion

In this thesis, we reviewed the fundamental principles of multistep methods, with a specific focus on the classical Nyström method for solving second-order initial value problems. We studied a High-Order Collocation-Based Nystrom Scheme for Second-Order IVPs, aiming to combine high numerical accuracy with computational efficiency. The method is constructed using a high-order Taylor series expansion and strategically selected collocation points, allowing for the systematic cancellation of lower-order error terms and yielding a global step point order of 8.

Optimized coefficients were derived to minimize local truncation errors, and the method was rigorously validated through a series of numerical experiments on both linear and nonlinear IVPs. The results confirmed that the proposed scheme is highly accurate, with significant error reduction observed as the step size decreases.

The formulation preserves the continuity of the solution and its derivatives up to the eighth order, ensuring reliability in approximating complex dynamical systems. Its high precision make it a valuable tool for applications in scientific computing and engineering.

This work also lays the groundwork for future exploration into more advanced strategies, theoretical refinements, and broader classes of differential equations where high-order accuracy is essential.

References

[1] M.K.Jain, S.R.K Iyengar, R.K.Jain. Numerical Methods for Scientific and Engineering Computation. New Age International (P) Limited, Publishers. New Delhi, 2012.

[2] Rufai MA, Mazzia F, Ramos H. An adaptive optimized Nystrom method for second-order IVPs. *Math Meth Appl Sci.* 2022;1-14. doi:10.1002/mma.8983

[3] Vander HouwenPJ, Sommeijer BP, Nguyen HC. Stability of collocation-based Runge-Kutta-Nystrom methods. BIT. 1991;31:469-481

MATLAB CODE:

```
% Constants and initial conditions
n = 40; % Number of steps
x0 = 0; % Initial x value
x_n = 1; % Final x value
h = (x_n - x0) / n; % Step size
y0 = -0.4;
dy_0 = -0.6;
y_exact=zeros(1,n+1);
for i=1:n+1
    y_exact(i) = (1/5) * exp(2*(x0+(i-1)*h)) * (sin(x0+(i-1)*h) -
        2*cos(x0+(i-1)*h));
end

% Define the functions for derivatives
d2y = @(x, y, dy) 2*dy-2*y+(sin(x))*(exp(2*x));
d3y = @(x, y, dy, d2y) exp(2*x)*cos(x) + 2*exp(2*x)*sin(x) - 2*dy + 2*d2y;
d4y = @(x, y, dy, d2y, d3y) 4*exp(2*x)*cos(x) + 3*exp(2*x)*sin(x) - 2*d2y +
    2*d3y;
d5y = @(x, y, dy, d2y, d3y, d4y) 11*exp(2*x)*cos(x) + 2*exp(2*x)*sin(x) -
    2*d3y + 2*d4y;
d6y = @(x, y, dy, d2y, d3y, d4y, d5y) 24*exp(2*x)*cos(x) - 7*exp(2*x)*sin(x)
    - 2*d4y + 2*d5y;
d7y = @(x, y, dy, d2y, d3y, d4y, d5y, d6y) 41*exp(2*x)*cos(x) -
    38*exp(2*x)*sin(x) - 2*d5y + 2*d6y;
d8y = @(x, y, dy, d2y, d3y, d4y, d5y, d6y, d7y)  44*exp(2*x)*cos(x) -
    117*exp(2*x)*sin(x) - 2*d6y + 2*d7y;

% Weights and constants
c1 = (7 - sqrt(21)) / 14;
c2 = 1/2;
c3 = (7 + sqrt(21)) / 14;

% Initialize arrays for results
x = linspace(x0, x_n, n + 1);
y = zeros(1, n + 1);
dy = zeros(1, n + 1);
error = zeros(1,n+1);
```

```
y(1) = y0;
dy(1) = dy_0;
fprintf('Initial conditions:\n');
fprintf('x = %.2f, y = %.20f, dy = %.5f, exact_y = %.20f, error = %.20e\n',
    ...
    x(1), y(1), dy(1), y_exact(1), abs(y(1) - y_exact(1)));

% Iterative computation
for i = 1:n
    xi = x(i); yi = y(i); dyi = dy(i);
    d2y_i = d2y(xi, yi, dyi);
    d3y_i = d3y(xi, yi, dyi, d2y_i);
    d4y_i = d4y(xi, yi, dyi, d2y_i, d3y_i);
    d5y_i = d5y(xi, yi, dyi, d2y_i, d3y_i, d4y_i);
    d6y_i = d6y(xi, yi, dyi, d2y_i, d3y_i, d4y_i, d5y_i);
    d7y_i = d7y(xi, yi, dyi, d2y_i, d3y_i, d4y_i, d5y_i, d6y_i);
    d8y_i = d8y(xi, yi, dyi, d2y_i, d3y_i, d4y_i, d5y_i, d6y_i, d7y_i);

    fc1 = d2y_i + h*d3y_i*c1 + (h^2/2)*d4y_i*c1^2 + (h^3/6)*d5y_i*c1^3 + ...
          (h^4/24)*d6y_i*c1^4 + (h^5/120)*d7y_i*c1^5 + (h^6/720)*d8y_i*c1^6;
    fc2 = d2y_i + h*d3y_i*c2 + (h^2/2)*d4y_i*c2^2 + (h^3/6)*d5y_i*c2^3 + ...
          (h^4/24)*d6y_i*c2^4 + (h^5/120)*d7y_i*c2^5 + (h^6/720)*d8y_i*c2^6;
    fc3 = d2y_i + h*d3y_i*c3 + (h^2/2)*d4y_i*c3^2 + (h^3/6)*d5y_i*c3^3 + ...
          (h^4/24)*d6y_i*c3^4 + (h^5/120)*d7y_i*c3^5 + (h^6/720)*d8y_i*c3^6;
    f1 = d2y_i + h*d3y_i + (h^2/2)*d4y_i + (h^3/6)*d5y_i + ...
         (h^4/24)*d6y_i + (h^5/120)*d7y_i + (h^6/720)*d8y_i;

    y(i+1) = yi + h*dyi + (h^2/360)*(18*d2y_i + 7*(sqrt(21)+7)*fc1 + ...
                64*fc2 - 7*(sqrt(21)-7)*fc3);
    dy(i+1) = dyi + (h/180)*(9*d2y_i + 49*fc1 + 64*fc2 + 49*fc3 + 9*f1);
    error(i) = abs(y(i) - y_exact(i));
    fprintf('Step %d: x = %.2f, y = %.20f, y_{exact} = %.20f, error =
       %.20e\n', ...
        i, x(i+1), y(i+1), y_exact(i+1), error(i));
end

% Plot the numerical and exact solutions
figure;
plot(x, y, 'b-*', 'LineWidth', 0.5); hold on;
```

```
plot(x, y_exact, 'r-o', 'LineWidth', 0.5);
xlabel('x'); ylabel('y(x)');
legend('Numerical Solution', 'Exact Solution');
grid on;

% Plot the error
figure;
plot(x, error, 'k', 'LineWidth', 2);
xlabel('x'); ylabel('Error |y_{num} - y_{exact}|');
grid on;

% Display results
MAE = max(error);
fprintf('\nMaximum Absolute Error (MAE): %.6e\n', MAE);
```

Listing 4.1: MATLAB code for solving Problem 3.3

```
n_coarse = 20;
n_fine = 2 * n_coarse;
function [x, y] = solver(n)
    x0 = 0; x_n = 2;
    h = (x_n - x0) / n;
    y0 = 2; dy_0 = 0;
    x = linspace(x0, x_n, n + 1);
    y = zeros(1, n + 1);
    dy = zeros(1, n + 1);
    y(1) = y0; dy(1) = dy_0;
    % Derivatives
    d2y = @(x, y, dy) (1 - y^2)*dy - y;
    d3y = @(x, y, dy, d2y) - (y^2 - 1)*d2y - 2*y*dy^2 - dy;
    d4y = @(x, y, dy, d2y, d3y) -2*dy^3 - (y^2 - 1)*d3y - d2y - 6*y*dy*d2y;
    d5y = @(x, y, dy, d2y, d3y, d4y) - (y^2 - 1)*d4y - 12*dy^2*d2y - ...
           6*y*d2y^2 - 8*y*dy*d3y - d3y;
    d6y = @(x, y, dy, d2y, d3y, d4y, d5y) - 20*dy^2*d3y - (y^2 - 1)*d5y - ...
           30*dy*d2y^2 - 10*y*dy*d4y - 20*y*d2y*d3y - d4y;
    d7y = @(x, y, dy, d2y, d3y, d4y, d5y, d6y) - (y^2 - 1)*d6y - ...
           20*y*d3y^2 - 30*d2y^3 - 30*dy^2*d4y - 120*dy*d2y*d3y - ...
           12*y*dy*d5y - 30*y*d2y*d4y - d5y;
    d8y = @(x, y, dy, d2y, d3y, d4y, d5y, d6y, d7y) - 42*dy^2*d5y - ...
           140*dy*d3y^2 - (y^2 - 1)*d7y - 210*d2y^2*d3y - ...
           14*y*dy*d6y - 42*y*d2y*d5y - 70*y*d3y*d4y - ...
           210*dy*d2y*d4y - d6y;
    c1 = (7 - sqrt(21)) / 14;
    c2 = 1/2;
    c3 = (7 + sqrt(21)) / 14;
    for i = 1:n
        xi = x(i); yi = y(i); dyi = dy(i);
        d2y_i = d2y(xi, yi, dyi);
        d3y_i = d3y(xi, yi, dyi, d2y_i);
        d4y_i = d4y(xi, yi, dyi, d2y_i, d3y_i);
        d5y_i = d5y(xi, yi, dyi, d2y_i, d3y_i, d4y_i);
        d6y_i = d6y(xi, yi, dyi, d2y_i, d3y_i, d4y_i, d5y_i);
        d7y_i = d7y(xi, yi, dyi, d2y_i, d3y_i, d4y_i, d5y_i, d6y_i);
        d8y_i = d8y(xi, yi, dyi, d2y_i, d3y_i, d4y_i, d5y_i, d6y_i, d7y_i);

        fc1 = d2y_i + h*d3y_i*c1 + (h^2/2)*d4y_i*c1^2 + ...
```

```
                (h^3/6)*d5y_i*c1^3 + (h^4/24)*d6y_i*c1^4 + ...
                (h^5/120)*d7y_i*c1^5 + (h^6/720)*d8y_i*c1^6;
        fc2 = d2y_i + h*d3y_i*c2 + (h^2/2)*d4y_i*c2^2 + ...
                (h^3/6)*d5y_i*c2^3 + (h^4/24)*d6y_i*c2^4 + ...
                (h^5/120)*d7y_i*c2^5 + (h^6/720)*d8y_i*c2^6;
        fc3 = d2y_i + h*d3y_i*c3 + (h^2/2)*d4y_i*c3^2 + ...
                (h^3/6)*d5y_i*c3^3 + (h^4/24)*d6y_i*c3^4 + ...
                (h^5/120)*d7y_i*c3^5 + (h^6/720)*d8y_i*c3^6;
        f1  = d2y_i + h*d3y_i + (h^2/2)*d4y_i + ...
                (h^3/6)*d5y_i + (h^4/24)*d6y_i + ...
                (h^5/120)*d7y_i + (h^6/720)*d8y_i;

        y(i+1) = yi + h*dyi + (h^2/360)*(18*d2y_i + ...
                    7*(sqrt(21)+7)*fc1 + 64*fc2 - ...
                    7*(sqrt(21)-7)*fc3);
        dy(i+1) = dyi + (h/180)*(9*d2y_i + 49*fc1 + ...
                    64*fc2 + 49*fc3 + 9*f1);
    end
end
[x_h, y_h] = solver(n_coarse);
[~, y_h2] = solver(n_fine);
y_h2_interp = y_h2(1:2:end);
p = 7;
error_dmp = abs(y_h - y_h2_interp) / (2^p - 1);
figure;
plot(x_h, y_h, 'b-*', 'DisplayName', 'n=20');
hold on;
x_h2 = linspace(0, 2, n_fine + 1);
plot(x_h2, y_h2, 'r-o', 'DisplayName', 'n=40');
legend('Location','Best');
xlabel('x'); ylabel('y(x)'); grid on;
figure;
plot(x_h, error_dmp, 'r', 'LineWidth', 2);
xlabel('x'); ylabel('Estimated Error'); grid on;
MAE = max(error_dmp);
fprintf('\nMax Estimated Error (Double Mesh): %.6e\n', MAE);
```

Listing 4.2: MATLAB code for solving problem 3.7 via Double Mesh Principle

```
% Define symbolic variables
syms y(x) x epsilon

% Given expression for y'' in terms of y and y'
dy = diff(y,x);
d2y = ((dy)^2)*(y);

% Compute higher-order derivatives
d3y = diff(d2y, x); % y'''
d4y = diff(d3y, x); % y''''
d5y = diff(d4y, x); % y'''''
d6y = diff(d5y, x); % y''''''
d7y = diff(d6y, x); % y'''''''
d8y = diff(d7y, x); % y''''''''

% Simplify expressions for clarity
d3y = simplify(d3y);
d4y = simplify(d4y);
d5y = simplify(d5y);
d6y = simplify(d6y);
d7y = simplify(d7y);
d8y = simplify(d8y);

% Display results
disp('d3y:')
disp(d3y)
disp('d4y:')
disp(d4y)
disp('d5y:')
disp(d5y)
disp('d6y:')
disp(d6y)
disp('d7y:')
disp(d7y)
disp('d8y:')
disp(d8y)
```

Listing 4.3: Symbolic computation of derivatives up to eighth order

```
% Define the known collocation points
c0 = 0; c1 = 1/2 - sqrt(21)/14; c2 = 1/2; c3 = 1/2 + sqrt(21)/14;
c4 = 1/2 + (2*sqrt(21))/21; c5 = 1;
% Define symbolic variables for r (integration variable) and f_cj
syms r f_c0 f_c1 f_c2 f_c3 f_c4 f_c5
% Define the Lagrange polynomials for j = 0, 1, 2, 3, 4, 5
L0 = (r - c1)*(r - c2)*(r - c3)*(r - c4)*(r - c5) / ...
    ((c0 - c1)*(c0 - c2)*(c0 - c3)*(c0 - c4)*(c0 - c5));
L1 = (r - c0)*(r - c2)*(r - c3)*(r - c4)*(r - c5) / ...
    ((c1 - c0)*(c1 - c2)*(c1 - c3)*(c1 - c4)*(c1 - c5));
L2 = (r - c0)*(r - c1)*(r - c3)*(r - c4)*(r - c5) / ...
    ((c2 - c0)*(c2 - c1)*(c2 - c3)*(c2 - c4)*(c2 - c5));
L3 = (r - c0)*(r - c1)*(r - c2)*(r - c4)*(r - c5) / ...
    ((c3 - c0)*(c3 - c1)*(c3 - c2)*(c3 - c4)*(c3 - c5));
L4 = (r - c0)*(r - c1)*(r - c2)*(r - c3)*(r - c5) / ...
    ((c4 - c0)*(c4 - c1)*(c4 - c2)*(c4 - c3)*(c4 - c5));
L5 = (r - c0)*(r - c1)*(r - c2)*(r - c3)*(r - c4) / ...
    ((c5 - c0)*(c5 - c1)*(c5 - c2)*(c5 - c3)*(c5 - c4));
% Now calculate the integral for each alpha_j (for s = 1)
alpha0 = int((1 - r) * L0, r, 0, 1);
alpha1 = int((1 - r) * L1, r, 0, 1);
alpha2 = int((1 - r) * L2, r, 0, 1);
alpha3 = int((1 - r) * L3, r, 0, 1);
alpha4 = int((1 - r) * L4, r, 0, 1);
alpha5 = int((1 - r) * L5, r, 0, 1);
% Now calculate the summation from j = 0 to j = 5 for alpha_j * f_cj
summation = alpha0*f_c0 + alpha1*f_c1 + alpha2*f_c2 + ...
            alpha3*f_c3 + alpha4*f_c4 + alpha5*f_c5;
% Simplify the result
summation = simplify(summation);
% Convert the result to decimal form
summation_decimal = vpa(summation, 6); % 6 decimal places
% Display the simplified decimal result
disp('The summation from j = 0 to j = 5 for alpha_j * f_cj (decimal form)
    is:');
disp(summation_decimal);
```

Listing 4.4: MATLAB code for formation of equation (2.1.6)

```
% Define the known collocation points
c0 = 0; c1 = 1/2 - sqrt(21)/14; c2 = 1/2; c3 = 1/2 + sqrt(21)/14;
c4 = 1/2 + (2*sqrt(21))/21; c5 = 1;
% Define symbolic variables for r (integration variable) and f_cj
syms r f_c0 f_c1 f_c2 f_c3 f_c4 f_c5
% Define the Lagrange polynomials for j = 0, 1, 2, 3, 4, 5
L0 = (r - c1)*(r - c2)*(r - c3)*(r - c4)*(r - c5) / ...
    ((c0 - c1)*(c0 - c2)*(c0 - c3)*(c0 - c4)*(c0 - c5));
L1 = (r - c0)*(r - c2)*(r - c3)*(r - c4)*(r - c5) / ...
    ((c1 - c0)*(c1 - c2)*(c1 - c3)*(c1 - c4)*(c1 - c5));
L2 = (r - c0)*(r - c1)*(r - c3)*(r - c4)*(r - c5) / ...
    ((c2 - c0)*(c2 - c1)*(c2 - c3)*(c2 - c4)*(c2 - c5));
L3 = (r - c0)*(r - c1)*(r - c2)*(r - c4)*(r - c5) / ...
    ((c3 - c0)*(c3 - c1)*(c3 - c2)*(c3 - c4)*(c3 - c5));
L4 = (r - c0)*(r - c1)*(r - c2)*(r - c3)*(r - c5) / ...
    ((c4 - c0)*(c4 - c1)*(c4 - c2)*(c4 - c3)*(c4 - c5));
L5 = (r - c0)*(r - c1)*(r - c2)*(r - c3)*(r - c4) / ...
    ((c5 - c0)*(c5 - c1)*(c5 - c2)*(c5 - c3)*(c5 - c4));
% Now calculate the integral for each beta_j (for s = 1)
beta0 = int(L0, r, 0, 1);
beta1 = int(L1, r, 0, 1);
beta2 = int(L2, r, 0, 1);
beta3 = int(L3, r, 0, 1);
beta4 = int(L4, r, 0, 1);
beta5 = int(L5, r, 0, 1);
% Now calculate the summation from j = 0 to j = 5 for beta_j * f_cj
summation = beta0*f_c0 + beta1*f_c1 + beta2*f_c2 + ...
            beta3*f_c3 + beta4*f_c4 + beta5*f_c5;
% Simplify the result
summation = simplify(summation);
% Convert the result to decimal form
summation_decimal = vpa(summation, 6); % 6 decimal places
% Display the simplified decimal result
disp('The summation from j = 0 to j = 5 for beta_j * f_cj (decimal form)
    is:');
disp(summation_decimal);
```

Listing 4.5: MATLAB code for formation of equation (2.1.7)

```
% Define symbolic variables
syms real  % Collocation points
c1 = 1/2 - sqrt(21)/14;
c2 = 1/2;
c3 = 1/2 + sqrt(21)/14;
c4 = 1/2 + (2*sqrt(21))/21;

% Define discrete collocation points I = {0, c1, c2, c3, c4, 1}
I = [0, c1, c2, c3, c4, 1];

% Define formula names
formulas = {'y_c1', 'y_c2', 'y_c3', 'y_c4', 'y1'};

% Define theta values for each formula
theta_values = {[-1, 1, 0, 0, 0, 0], [-1, 0, 1, 0, 0, 0], ...
                [-1, 0, 0, 1, 0, 0], [-1, 0, 0, 0, 1, 0], ...
                [-1, 0, 0, 0, 0, 1]};

% Define capital Theta values
Theta_values = {[(7 - sqrt(21)) / 14, 0, 0, 0, 0, 0], [1/2, 0, 0, 0, 0, 0],
    ...
                [(7 + sqrt(21)) / 14, 0, 0, 0, 0, 0], [1/2 + 2/sqrt(21), 0,
                   0, 0, 0, 0], ...
                [1, 0, 0, 0, 0, 0]};

% Define varphi values for each formula
varphi_c1 = [ (1111 - 171*sqrt(21)) / 41160, 55 / 6174, ...
              2*(419 - 98*sqrt(21)) / 15435, (1709 - 343*sqrt(21)) / 17640,
                 ...
              -18 / 1715, (12*sqrt(21) + 41) / 20580 ];

varphi_c2 = [ 49 / 1920, (7 * (8*sqrt(21) + 35)) / 5760, ...
              1 / 72, -(7 * (8*sqrt(21) - 35)) / 5760, ...
              0, 1 / 1920 ];

varphi_c3 = [ (171*sqrt(21) + 919) / 41160, (2401*sqrt(21) + 11003) /
    123480, ...
              2 * (98*sqrt(21) + 449) / 15435, 1 / 882, ...
              18 / 1715, - (12*sqrt(21) + 55) / 20580 ];
```

```
36
37 varphi_c4 = [ (88612*sqrt(21) + 426493) / 17781120, ((49*sqrt(7/3)) / 720) +
       (5545181 / 53343360), ...
38                 (129361 / 1666980) + (16 / (45 * sqrt(21))), ((7*sqrt(7/3)) /
                       720) + (112019 / 7620480), ...
39                 25 / 24696, -(788*sqrt(21) + 3607) / 3556224 ];
40
41 varphi_y1 = [ 1 / 20, (7 / 360 )* (sqrt(21) + 7), 8 / 45, (1 / 360 )* (-7 *
       (sqrt(21) - 7)), 0, 0 ];
42
43 varphi_values = {varphi_c1, varphi_c2, varphi_c3, varphi_c4, varphi_y1};
44
45 % Initialize order results
46 order_results = struct();
47
48 % Loop through each formula
49 for i = 1:length(formulas)
50
51     theta_j = theta_values{i};
52     Theta_j = Theta_values{i};
53     varphi_j = varphi_values{i};
54
55     % Initialize order search
56     q = 0;
57     C_q = 0;
58
59     % Iterate until C_q is nonzero
60     while C_q == 0
61         % Compute summation terms for C_q using Equation (13)
62         sum_theta = sum(I.^q .* theta_j);
63
64         % Handle q=0 separately to avoid division by zero
65         if q == 0
66             sum_Theta = 0;
67             sum_varphi = 0;
68         elseif q == 1
69             sum_Theta = q * sum(I.^(q-1) .* Theta_j);
70             sum_varphi = 0;  % Since (q-2) is negative
71         else
```

```
            sum_Theta = q * sum(I.^(q-1) .* Theta_j);
            sum_varphi = q * (q-1) * sum(I.^(q-2) .* varphi_j);
        end

        % Compute C_q
        C_q = (1 / factorial(q)) * (sum_theta - sum_Theta - sum_varphi);

        % If C_q is still zero, increase q
        if abs(C_q) < 1e-10  % Tolerance for numerical errors
            C_q = 0;
            q = q + 1;
        else
            break;
        end
    end

    % Store LTE order
    order_results.(formulas{i}) = q;
end

% Display results
disp('Computed LTE Orders:');
disp(order_results);
```

Listing 4.6: MATLAB code for finding Order(LTE) of $y_{c_1}, y_{c_2}, y_{c_3}, y_{c_4}, y_1$ using (2.2.5)

```
% Define symbolic variables
syms real

% Collocation points
c1 = 1/2 - sqrt(21)/14;
c2 = 1/2;
c3 = 1/2 + sqrt(21)/14;
c4 = 1/2 + (2*sqrt(21))/21;

% Define discrete collocation points I = {0, c1, c2, c3, c4, 1}
I = [0, c1, c2, c3, c4, 1];

% Define formula names for derivatives
formulas_p = {'y_p_c1', 'y_p_c2', 'y_p_c3', 'y_p_c4', 'y_p_1'};

% Extract theta values from bottom 5 rows of S (Equation 9)
theta_p_values = {
    [0, 0, 0, 0, 0, 0],...
    [0, 0, 0, 0, 0, 0],  ...
    [0, 0, 0, 0, 0, 0],  ...
    [0, 0, 0, 0, 0, 0],  ...
    [0, 0, 0, 0, 0, 0]
};

% Extract Theta values from bottom 5 rows of S (Equation 9)
Theta_p_values = {
    [1, -1, 0, 0, 0, 0],  ...
    [1, 0, -1, 0, 0, 0],  ...
    [1, 0, 0, -1, 0, 0],  ...
    [1, 0, 0, 0, -1, 0],  ...
    [1, 0, 0, 0, 0, -1]
};

% Extract varphi values from bottom 5 rows of T (Equation 10)
varphi_p_values = {
    [(41*sqrt(21) + 189) / 5880,  (49/360)  - 23/(840*sqrt(21)), ...
     (8/45) - 106/(105*sqrt(21)),   (343 - 29*sqrt(21)) / 2520, ...
     - (6 * sqrt(3/7)) / 35,  (1/56 ) + 41/(280*sqrt(21))],...

```

```
    [(81 - 8*sqrt(21)) / 960,    (95*sqrt(21) + 392) / 2880, ...
     (sqrt(7/3)/48) + (8/45), - (7 * (25*sqrt(21) - 56)) / 2880, ...
     (3*sqrt(21)) / 80,  (-8*sqrt(21) - 33) / 960],...

    [(23*sqrt(21) + 189) / 5880,  (49/360) + 523/(840*sqrt(21)), ...
     (8/45) + 86/(105*sqrt(21)), (7/360) * (sqrt(21) + 7), ...
     - (6 * sqrt(3/7)) / 35,   1/56 + 23/(280*sqrt(21))],...

    [(3368*sqrt(21) + 48027) / 1270080, (49/360) + 113269/(181440*sqrt(21)),
       ...
     (8/45) + 36871/(45360*sqrt(21)), (49/360) + 16717/(25920*sqrt(21)), ...
     -65/(336*sqrt(21)),  (737/60480) + 421/(7560*sqrt(21))],...

    [1/20,   49/180, 16/45,  49/180 , 0, 1/20 ]
};

% Initialize order results for derivatives
order_p_results = struct();

% Loop through each formula for y'_c1, y'_c2, ..., y'_1
for i = 1:length(formulas_p)

    theta_p_j = theta_p_values{i};
    Theta_p_j = Theta_p_values{i};
    varphi_p_j = varphi_p_values{i};

    % Initialize order search
    q = 0;
    C_q = 0;

    % Iterate until C_q is nonzero
    while C_q == 0
        % Compute summation terms for C_q using Equation (13)
        sum_theta_p = sum(I.^q .* theta_p_j);

        % Handle q=0 separately
        if q == 0
            sum_Theta_p = 0;
            sum_varphi_p = 0;
```

```
        elseif q == 1
            sum_Theta_p = q * sum(I.^(q-1) .* Theta_p_j);
            sum_varphi_p = 0;  % (q-2) negative, ignore
        else
            sum_Theta_p = q * sum(I.^(q-1) .* Theta_p_j);
            sum_varphi_p = q * (q-1) * sum(I.^(q-2) .* varphi_p_j);
        end

        % Compute C_q
        C_q = (1 / factorial(q)) * (sum_theta_p - sum_Theta_p -
            sum_varphi_p);

        % Increase q until non-zero C_q found
        if abs(C_q) < 1e-10
            C_q = 0;
            q = q + 1;
        else
            break;
        end
    end

    % Store LTE order for y'_ci
    order_p_results.(formulas_p{i}) = q - 1;
end

% Display results
disp('Computed LTE Orders for Derivatives:');
disp(order_p_results);
```

Listing 4.7: MATLAB code for finding Order(LTE) of $y'_{c_1}, y'_{c_2}, y'_{c_3}, y'_{c_4}, y'_1$ using (2.2.5)